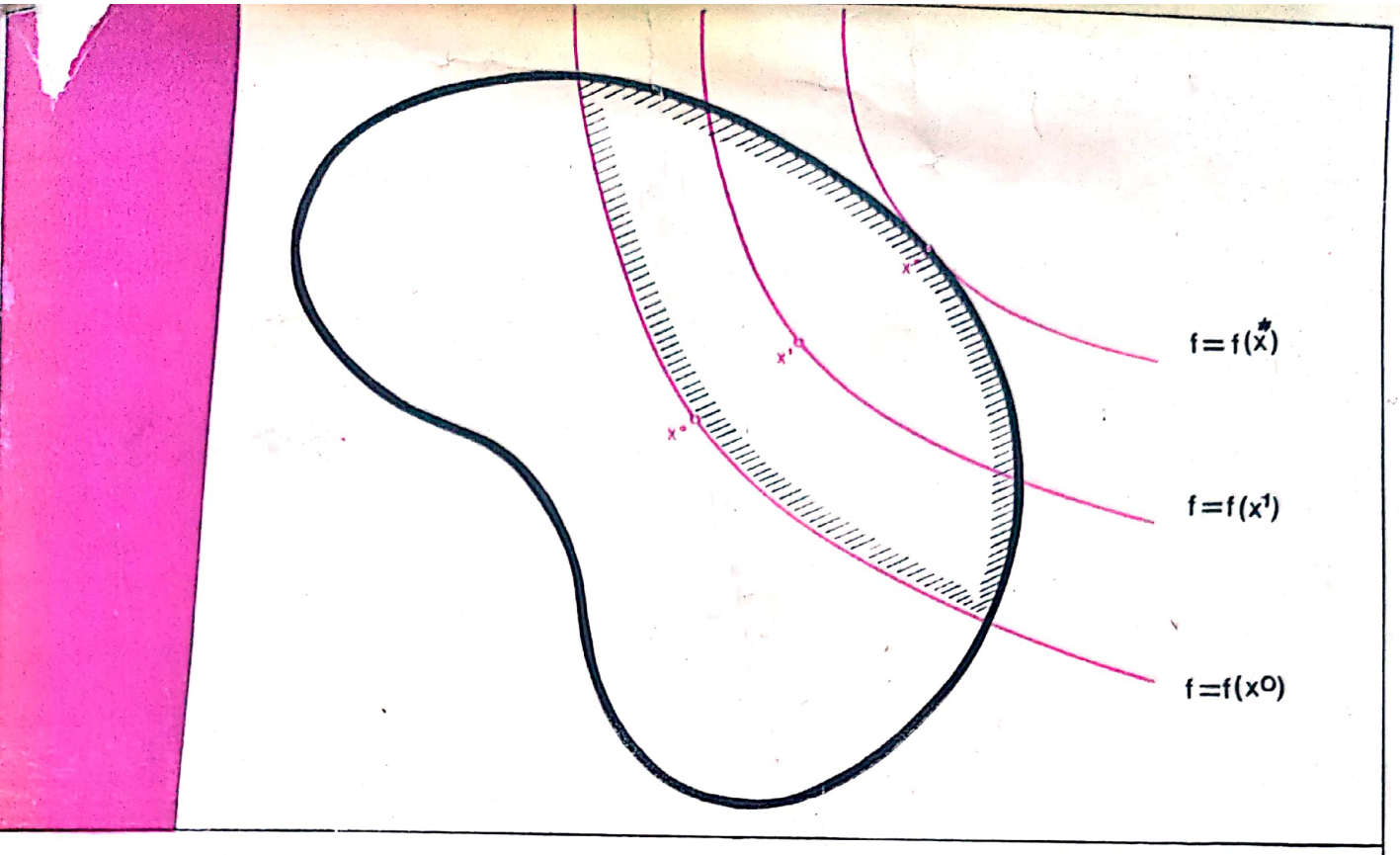

IOAN MARUŞCIAC

METODE DE REZOLVARE A PROBLEMELOR DE PROGRAMARE NELINIARĂ

EDITURA DACIA

Scanned with OKEN Scanner

SERIA

LUCRĂRILE SEMINARULUI DE CEA MAI BUNĂ APROXIMAȚIE ȘI PROGRAMARE MATEMATICĂ

Vol. II

Seminarul de cea mai bună aproximație și programare matematică își desfășoară activitatea la Institutul de calcul, Filiala Cluj a Academiei R. S. România.

Scanned with OKEN Scanner

IOAN MARUŞCIAC

METODE DE REZOLVARE A PROBLEMELOR DE PROGRAMARE NELINIARĂ

EDITURA DACIA
CLUJ 1973

Scanned with OKEN Scanner

CAPITOLUL I

PROBLEMA GENERALĂ DE PROGRAMARE MATEMATICĂ

§ 1. Introducere

Capitolul de programare matematică, cu toate că este un capitol relativ nou al matematicii, datorită multiplelor aplicaţii pe care le are, s-a dezvoltat într-un ritm extrem de mare. De aceea ar fi foaret greu de sintetizat într-o lucrare de proporţiile acesteia toate rezultatele mai importante obţinute în acest domeniu. În prezenta lucrare am încercat să redăm cîteva dintre principalele rezultate în rezolvarea problemelor de programare neliniară. Pentru a nu lungi expunerea unele rezultate sînt prezentate fără demonstraţie. S-a urmărit însă în mod consecvent ca algoritmii să fie prezentaţi de aşa manieră încît cititorul să fie în stare să-i înveţe şi apoi să-i aplice efectiv. Acesta a fost motivul pentru care s-a dat şi cîte un exemplu numeric aproape după fiecare algoritm prezentat.

În capitolul I, după un scurt istoric al problemei, se prezintă pe scurt teoremele lui KUHN–TUCKER, care stau la baza multor criterii de optim în programare neliniară.

Datorită faptului că o mare parte din algoritmii prezentaţi constituie o aplicare succesivă a metodei simplex, în ultima parte a capitolului se prezintă metoda eliminării a lui JORDAN şi metoda simplex într-o variantă mai puţin cunoscută, dată de STIEFEL şi ZUHOVIŢKI.

În capitolul II se trec în revistă principalele metode de rezolvare a problemelor de programare neliniară. Capitolul III este consacrat cazurilor speciale de probleme de programare neliniară. Printre altele aici se prezintă problema de programare segmentar liniară şi programarea monotonă.

Capitolul IV conţine metode de rezolvare a problemelor de programare pătratică. În afară de algoritmii cunoscuţi pentru programarea pătratică definită, se dau şi unele metode de rezolvare a programelor nedefinite.

Ultimul capitol este consacrat programării simultane a mai multor funcţii de scop. După ce se prezintă un algoritm pentru rezolvarea unei asemenea probleme în care funcţiile de scop sînt pătratice şi se cere minimizarea maximului lor, se tratează problema optimizării simultane a mai multor funcţii de scop în raport cu o anumită relaţie de ordine din spaţiul valorilor funcţiilor de scop.

Scanned with OKEN Scanner

§ 2. Istoricul problemei

În programarea matematică se studiază, după cum se știe, o problemă importantă pentru practică, aceea a optimizării (maximizării sau minimizării) unei funcții liniare sau neliniare în condițiile existenței unor restricții suplimentare la care sînt supuse variabilele independente. Întrucît aceste restricții sînt date de obicei sub forma unor inegalități, metodele clasice din analiza matematică pentru calculul extremelor cu legături s-au dovedit a fi ineficiente.

Formularea problemei generale de programare liniară a fost făcută pentru prima dată în anul 1939 de către KANTOROVICI L.V. [31], care a prezentat o metodă foarte eficace, numită metoda „multiplicatorilor rezolvanți", cu care se poate rezolva problema generală de programare liniară.

Trebuie menționat că apariția problemei de programare liniară a dat un nou impuls și altor cercetări matematice legate de programarea liniară. Au căpătat o nouă importanță unele lucrări clasice, ca cele ale lui FARKAS, J. [18], MOTZKIN, T. S. [62] legate de teoria inegalităților liniare precum și lucrările lui NEUMANN J. VON [64] despre teoria jocurilor.

O importanță deosebită pentru programarea liniară a constituit-o *metoda simplex*, elaborată în 1949 de către DANTZIG G. B. și prezentată în vara anului 1949 la prima conferință de matematici economice organizată la Chicago cu tema „Cowles Commission for Research in Economics".

De atunci cercetările din acest domeniu s-au dezvoltat foarte mult, așa încît astăzi programarea liniară constituie un capitol, teoretic chiar, foarte important al matematicii moderne.

Mai menționăm că metoda simplex s-a dovedit extrem de utilă și pentru rezolvarea altor probleme de matematică strîns legate de programarea matematică. Printre acestea amintim problema clasică de aproximare uniformă (cebîșeviană) a unui sistem incompatibil de ecuații liniare.

Programarea neliniară studiază o problemă, de asemenea importantă pentru practică, de găsirea optimului unei funcții neliniare în care variabilele sînt supuse unor restricții în general neliniare, care determină un anumit domeniu din spațiul euclidian n-dimensional. Un caz important al acestei probleme îl constituie *problema de programare convexă* în care funcția de optimizat este o funcție convexă, iar inegalitățile determină un domeniu convex.

Problema de programare neliniară a fost pentru prima dată formulată în anul 1950 de către KUHN H. W. și TUCKER W. [40], care au prezentat lucrarea intitulată: „Programarea neliniară" (Nonlinear Programming), la simpozionul de probabilități și statistică matematică (Symposium on Mathematical Statistics and Probabilities), ținut în vara anului 1950 la Berkeley. În această lucrare autorii au pus bazele teoretice ale programării neliniare prin acele teoreme de punct șa care le poartă numele. De atunci și pînă acum această problemă a fost studiată de foarte mulți matematicieni. Au apărut între timp peste 500 de lucrări științifice consacrate acestei probleme. Majoritatea acestor lucrări au însă un caracter teoretic și numai

Scanned with OKEN Scanner

un număr relativ restrîns de lucrări sînt consacrate elaborării unor metode practice pentru rezolvarea diverselor tipuri de probleme de programare neliniară.

Ca şi problema de programare liniară, problema de programare neliniară apare de asemenea în foarte multe domenii de activitate economică. Aceasta apare de obicei atunci cînd în construirea unui model matematic se iau în considerare mai mulţi factori. Astfel, de exemplu, problema transportului, care este o problemă liniară, devine neliniară, dacă se consideră că preţul de cost al unei cantităţi transportate depinde şi de volumul total transportat. De asemenea, problemele de programare liniară devin neliniare dacă se ţine seamă şi de un anumit risc şi nesiguranţă care pot interveni în procesul de producţie. Din acest tip de probleme face parte, de exemplu, problema de programare stohastică.

Menţionăm de asemenea că de problema de programare neliniară este strîns legată problema de aproximare uniformă a unei funcţii de variabilă complexă. Un exemplu de acest fel îl constituie rezolvarea aproximativă a unui sistem incompatibil de ecuaţii liniare complexe, fie cazul în care necunoscutele nu sînt supuse altor restricţii, sau cel în care sînt supuse unor restricţii sub forma unor inegalităţi, care determină un domeniu convex.

§ 3. Formularea problemei generale de programare neliniară şi ideile lui Kuhn şi Tucker

Problema generală de programare matematică se enunţă astfel: *să se găsească minimul (maximul) funcţiei de scop*

$$f(\mathbf{x}) = f(x_1, x_2, \ldots, x_n) \tag{3.1}$$

în condiţiile

$$g_j(\mathbf{x}) = g_j(x_1, x_2, \ldots, x_n) \leqslant 0, \qquad (j = 1, 2, \ldots, m), \tag{3.2}$$

unde $f, g_j : \mathbf{R}^n \to \mathbf{R}$, $j = 1, 2, \ldots, m$.

Uneori problema generală de programare neliniară se enunţă sub formă puţin modificată, numită şi *forma canonică*, în felul următor: *să se găsească minimul (maximul) funcţiei de scop (3.1) în condiţiile (3.2) la care se mai adaugă condiţiile de nenegativitate a variabilelor:*

$$\mathbf{x} \geqslant \mathbf{0}, \text{ adică} \qquad x_i \geqslant 0, \qquad (i = 1, 2, \ldots, n). \tag{3.3}$$

În cazul bidimensional problema de programare neliniară este ilustrată în Figura 3.1, unde $\mathbf{x} = (x_1, x_2)$.

Aici inegalităţile (3.2) şi (3.3) determină un domeniu plan Ω, care a fost haşurat, $f(\mathbf{x}) = \text{const.}$ reprezintă ecuaţia unei linii de nivel. Soluţia

Scanned with OKEN Scanner

problemei este punctul $\mathbf{x}^*$ în care o linie de nivel este tangentă la curba $g_1(\mathbf{x}) = 0$ (am considerat aici problema de minim pentru funcţia $f(\mathbf{x})$).

Astăzi încă sîntem departe de rezolvarea algoritmică completă a unei asemenea probleme generale de programare neliniară.

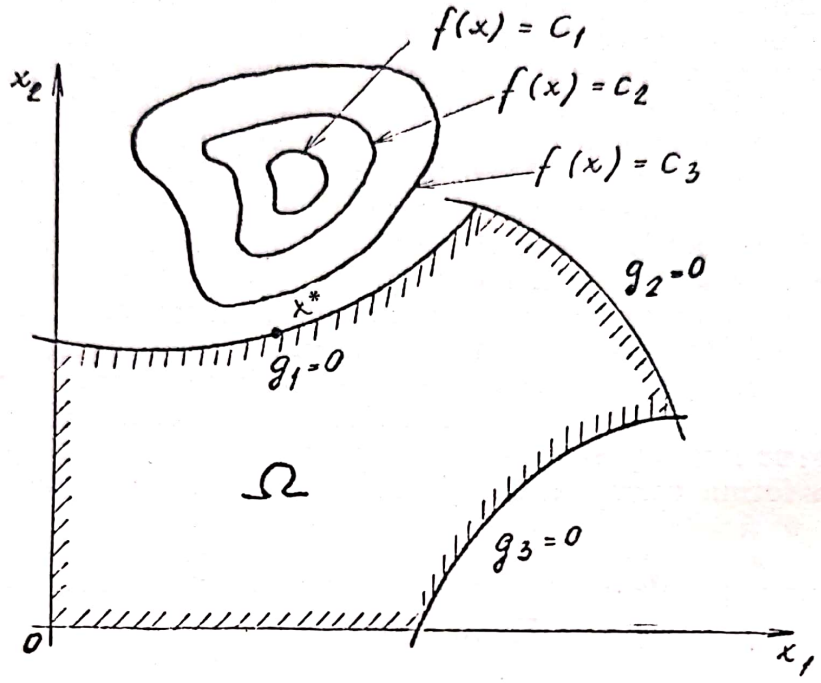

Figura 3.1.

Pentru a ajunge la construirea unui algoritm pentru o asemenea problemă este necesar să se pună condiţii în plus atît asupra funcţiei de scop cît şi pentru funcţiile-restricţie. Vom ajunge astfel la problema de programare convexă sau concavă.

Vom începe mai întîi cu problema clasică a minimului cu legături. Astfel ne punem problema să găsim *minimul funcţiei*

$$f(\mathbf{x}) = f(x_1, x_2, \ldots, x_n)$$

cînd variabilele $x_1, x_2, \ldots, x_n$ *verifică condiţiile*

$$g_j(\mathbf{x}) = g_j(x_1, x_2, \ldots, x_n) = 0, \qquad (j = 1, 2, \ldots, m<n). \tag{3.4}$$

Vom presupune că funcţiile f şi g_j, $j = 1, 2, \ldots, m$, admit derivate parţiale de ordinul întîi în raport cu toate variabilele x_i, $i = 1, 2, \ldots, n$.

Atunci utilizînd metoda multiplicatorilor lui Lagrange, considerăm funcţia auxiliară. $F : \mathbf{R}^{n+m} \to \mathbf{R}$,

$$F(\mathbf{x}, \mathbf{u}) = f(\mathbf{x}) + \sum_{j=1}^{m} u_j g_j(\mathbf{x}) \tag{3.5}$$

în care $u_1, u_2, \ldots, u_m$ sînt multiplicatorii lui Lagrange.

Scanned with OKEN Scanner

Condiția necesară de minim a funcției (3.5) constă în anularea derivatelor parțiale de ordinul întîi, adică

$$\frac{\partial F}{\partial x_i} = \frac{\partial f}{\partial x_i} + \sum_{j=1}^{m} u_j \frac{\partial g_j}{\partial x_i} = 0, \quad (i = 1, 2, \ldots, n) \tag{3.6}$$

$$\frac{\partial F}{\partial u_j} = g_j = 0, \quad (j = 1, 2, \ldots, m). \tag{3.7}$$

Ecuațiile (3.6) — (3.7) ne dau $m + n$ relații pentru cele $m + n$ necunoscute. Natural, condițiile (3.6) — (3.7) sînt doar condiții necesare de minim și nu și suficiente. În anumite condiții suplimentare pentru f și g_j aceste condiții devin și suficiente.

De exemplu, dacă funcția f este convexă iar restricțiile g_j sînt toate liniare, atunci se poate verifica ușor că într-adevăr condițiile (3.6) — (3.7) sînt și suficiente pentru minim.

Ne întoarcem acum la problema inițială de programare neliniară, adică la problema: *să se minimizeze funcția*

$$f(\mathbf{x}) = f(x_1, x_2, \ldots, x_n) \tag{3.8}$$

în condițiile:

$$g_j(\mathbf{x}) = g_j(x_1, x_2, \ldots, x_n) \leqslant 0, \quad (j = 1, 2, \ldots, n). \tag{3.9}$$

Considerăm acum m variabile reale auxiliare w_j astfel încît

$$g_j(\mathbf{x}) + w_j^2 = 0, \quad (j = 1, 2, \ldots, m). \tag{3.10}$$

Atunci problema de minim (3.8) — (3.9) se poate formula astfel: *să se minimizeze* $f(x_1, x_2, \ldots, x_n)$ *cînd variabilele* $x_1, x_2, \ldots, x_n, w_1, w_2, \ldots, w_m$ *verifică condițiile* (3.10).

Funcția lui Lagrange corespunzătoare acestei probleme de minim este

$$F(\mathbf{x}, \mathbf{w}, \mathbf{u}) = f(\mathbf{x}) + \sum_{j=1}^{m} u_j [g(\mathbf{x}) + w_j^2]. \tag{3.11}$$

Condiția necesară de minim se scrie în acest caz sub forma

$$\frac{\partial F}{\partial x_i} = \frac{\partial f}{\partial x_i} + \sum_{j=1}^{m} u_j \frac{\partial g_j}{\partial x_i} = 0, \quad (i = 1, 2, \ldots, n) \tag{3.12}$$

$$\frac{\partial F}{\partial u_j} = g_j + w_j^2 = 0, \quad (j = 1, 2, \ldots, m) \tag{3.13}$$

$$\frac{\partial F}{\partial w_j} = 2u_j w_j = 0, \quad (j = 1, 2, \ldots, m). \tag{3.14}$$

Scanned with OKEN Scanner

Deoarece w_j sînt variabilele ajutătoare, le vom elimina din sistemul (3.12) — (3.13). Astfel din (3.13) rezultă că oricare ar fi $j = 1, 2, \ldots, m$; $g_j = 0$ are loc atunci şi numai atunci cînd $w_j = 0$. De aceea, ţinînd seamă de (3.14) are loc condiţia

$$u_j g_j = 0, \quad (j = 1, 2, \ldots, m). \tag{3.14'}$$

Eliminînd variabilele w_j de asemenea şi din celelalte relaţii (3.12)—(3.13), rezultă următoarele condiţii pentru minim:

$$\frac{\partial f}{\partial x_i} + \sum_{j=1}^{m} u_j \frac{\partial g_j}{\partial x_i} = 0, \quad (i = 1, 2, \ldots, n) \tag{3.15}$$

$$g_j \leqslant 0, \quad (j = 1, 2, \ldots, m) \tag{3.16}$$

$$u_j g_j = 0, \quad (j = 1, 2, \ldots, m). \tag{3.17}$$

Evident că şi condiţiile (3.15) — (3.17) sînt doar condiţii *necesare* de minim. KUHN şi TUCKER [40] au arătat că, pentru ca funcţia f să aibă un minim pe punctul $\mathbf{x}$, este necesar ca în afara condiţiilor (3.15 — (3.17) să avem în plus condiţia

$$u_j \geqslant 0, \quad (j = 1, 2, \ldots, m). \tag{3.18}$$

În cazul maximului inegalităţile în (3.18) se înlocuiesc cu

$$u_j \leqslant 0, \quad (j = 1, 2, \ldots, m). \tag{3.18'}$$

Condiţiile (3.15) — (3.18), numite condiţiile lui Kuhn-Tucker, sînt condiţii necesare de minim atît pentru minim local cît şi pentru cel global. Kuhn şi Tucker au demonstrat că aceste condiţii devin şi suficiente cînd funcţiile f şi g_j sînt *funcţii convexe*. Prin funcţie convexă pe $D \subset \mathbf{R}^n$ se înţelege o funcţie convexă în sensul definiţiei lui Jensen:

$$\forall \mathbf{x}', \mathbf{x}'' \in D, \ \forall \lambda \in [0,1] \Rightarrow f(\lambda \mathbf{x}' + (1 - \lambda)\mathbf{x}'') \leqslant \lambda f(\mathbf{x}') + (1 - \lambda) f(\mathbf{x}'').$$

Prin urmare, în cazul problemei de programare convexă, condiţiile (3.15) — (3.18) sînt condiţii necesare şi suficiente pentru soluţia problemei.

Vom schiţa acum o metodă bazată pe traiectorii ortogonale ale unei familii de curbe (respectiv suprafeţe), care permite găsirea punctului optim (care realizează minimul funcţiei de scop), pornind de la un punct arbitrar din spaţiul $\mathbf{R}^n$ sau respectiv din domeniul admisibil Ω.

Vom începe mai întîi cu cazul minimului unei funcţii, fără restricţii.

Fie deci f o funcţie convexă. Pentru ilustrare vom considera cazul unei funcţii de două variabile x_1 şi x_2: $z = f(x_1, x_2)$. Liniile de nivel

$(x_1, x_2) = const.$ (Fig. 3.2) sînt niște curbe închise care mărginesc un domeniu convex. Pornind de la un punct P_0 al unei linii de nivel, se ajunge cel mai rapid la punctul de minim dacă deplasarea se face de-a lungul unei traiectorii ortogonale la familia liniilor de nivel.

După cum se știe ecuația diferențială a acestor traiectorii ortogonale este

$$\frac{d\mathbf{x}}{dt} = -\nabla f(\mathbf{x}), \tag{3.19}$$

unde

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_n}\right)$$

este gradientul funcției f în punctul $\mathbf{x}$.

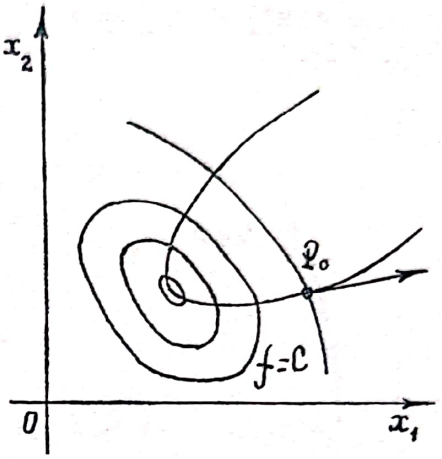

Figura 3.2.

Soluția ecuației (3.19) este o vector-funcție care depinde de două constante arbitrare și de parametrul t. Determinînd constantele din condițiile inițiale ale lui Cauchy, vector-funcția $\mathbf{x}(t)$ reprezintă o curbă plană — traiectoria ortogonală care trece prin punctul P_0. Traiectoria însăși interesează mai puțin aici. Este importantă doar limita lui $\mathbf{x}(t)$ pentru $t \to \infty$, întrucît se arată că [42] $\lim_{t\to\infty} \mathbf{x}(t) = \mathbf{x}^0$, unde $\mathbf{x}^0$ este punctul de minim căutat.

Să considerăm acum problema noastră de optim pentru funcția $f(\mathbf{x})$ cu restricțiile

$$g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots, m).$$

Atunci ecuația diferențială a traiectoriilor ortogonale (3.19) se schimbă în ecuația

$$\frac{d\mathbf{x}}{dt} = -\nabla f(\mathbf{x}) - \sum_{j=1}^{m} h_j(\mathbf{x}) \nabla g_j(\mathbf{x}) \tag{3.20}$$

unde

$$h_j(\mathbf{x}) = \begin{cases} 0, & \text{dacă } g_j(\mathbf{x}) \leqslant 0 \\ M, & \text{dacă } g_j(\mathbf{x}) > 0 \end{cases}$$

M fiind o constantă pozitivă suficient de mare.

Scanned with OKEN Scanner

Presupunînd că ∇f şi ∇g_j, $j = 1, 2, \ldots, m$, sînt continui, atunci ecuaţia (3.20) are o soluţie $\mathbf{x}(t)$, pentru care

$$\lim_{t\to\infty} \mathbf{x}(t) = \mathbf{x}^*$$

în care $g_j(\mathbf{x}^*) \leqslant 0$, $j = 1, 2, \ldots, m$. Deci $\mathbf{x}^*$ este soluţia optimă căutată (Fig. 3.3).

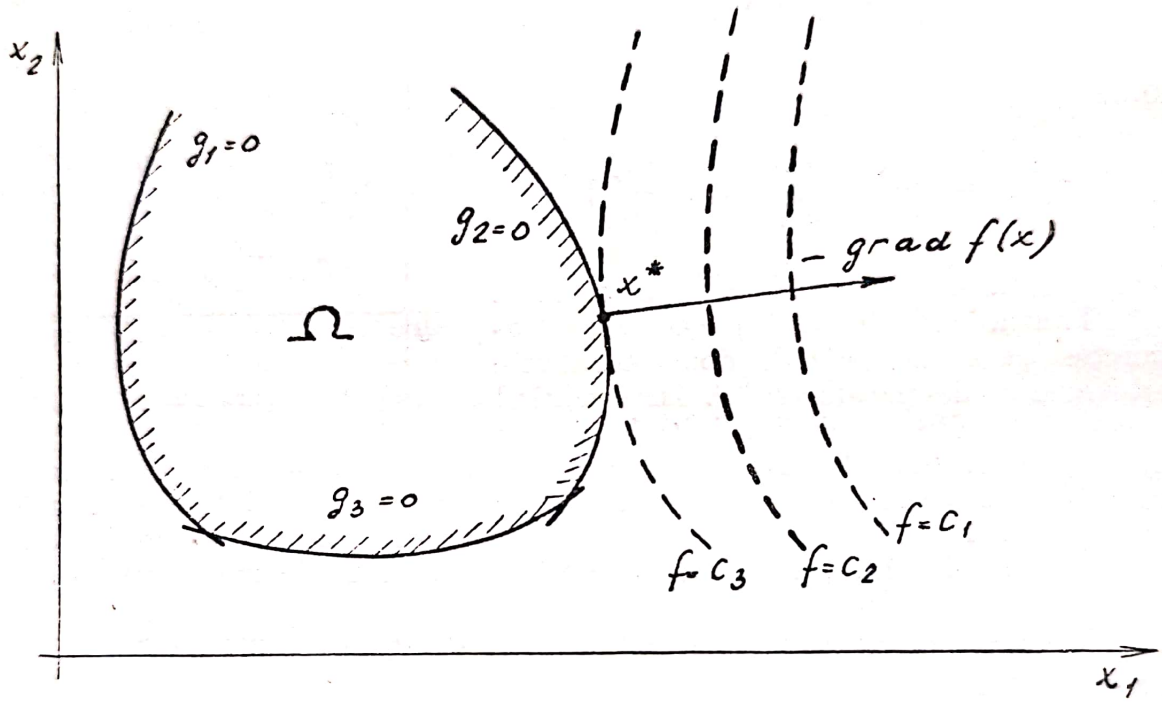

Figura 3.3.

Dacă $g_j(\mathbf{x}^*) < 0$, atunci $h_j[\mathbf{x}(t)] = 0$, pentru t suficient de mare. Dacă însă $g_j(x^*) = 0$ atunci $h_j[\mathbf{x}(t)]$ oscilează între 0 şi M şi astfel $\mathbf{x}(t)$ va fi de asemenea o funcţie oscilantă în jurul punctului $\mathbf{x}^*$.

Considerăm acum pentru fiecare $j = 1, 2, \ldots, m$, valoarea medie

$$u_j^* = \lim_{T\to\infty} \frac{1}{T} \int_T^{2T} h_j[\mathbf{x}(t)]dt.$$

Dacă integrăm membru cu membru relaţia (3.20) ce reprezintă ecuaţia traiectoriilor ortogonale, obţinem

$$\mathbf{x}(2T) - \mathbf{x}(T) = \int_T^{2T} -\nabla f(\mathbf{x}(t))dt - \int_T^{2T} \sum_{j=1}^{m} h_j(\mathbf{x}(t))\nabla g_j(\mathbf{x}(t))dt.$$

Împărţind cei doi membri cu T şi trecînd apoi la limită pentru $T \to \infty$, se demonstrează (vezi WOLFE, PH. [79]) că membrul întîi va tinde către zero şi astfel vom obţine relaţia

$$0 = -\nabla f(\mathbf{x}^*) - \sum_{j=1}^{m} u_j^* \nabla g_j(\mathbf{x}^*),$$

sau relaţia

$$-\nabla f(\mathbf{x}^*) = \sum_{j=1}^{m} u_j^* \nabla g_j(\mathbf{x}^*), \quad u_j^* \geqslant 0, \tag{3.21}$$

unde $u_j^* = 0$, dacă $g_j(\mathbf{x}^*) < 0$.

Relaţia (3.21) permite enunţarea următorului criteriu de optim.

TEOREMA 3.1. *Într-un punct optimal* $\mathbf{x}^*$ *al problemei de programare neliniară* (3.8) — (3.9) *gradientul cu semn schimbat al funcţiei de scop* f *este o combinaţie liniară cu coeficienţi nenegativi a normalelor suprafeţelor de restricţie ce trec prin punctul* $\mathbf{x}^*$.

Menţionăm că această metodă este descrisă detaliat în cartea lui KÜNZI H. P. şi KRELLE W. [42] precum şi în cartea lui WOLFE, PH. [78] care tratează exclusiv problema de programare neliniară. De exemplu în cartea lui Künzi şi Krelle, pe lîngă o serie de consideraţii teoretice sînt prezentate şi un număr de zece metode dintre cele mai importante care permit rezolvarea diverselor tipuri de probleme de programare neliniară. Unele dintre ele vor fi descrise şi în prezenta lucrare.

§ 4. Metoda eliminării a lui Jordan, cu aplicaţii

Este bine cunoscut faptul că matematica a pătruns în cele mai variate domenii ale activităţii umane. Din această cauză uneori sînt puse în situaţia să folosească diverse metode matematice şi persoane fără o pregătire matematică specială. Or, pentru ca metodele matematice să poată fi utilizate şi de nespecialişti este necesar ca ele să fie prezentate sub forma unor scheme şi reguli mecanice uşor de înţeles, fără să se facă apel la alte cunoştinţe matematice. Tocmai din această cauză în majoritatea cărţilor şi revistelor din lume cu caracter aplicativ, metodele matematice propuse sînt prezentate sub forma unor scheme şi tabele, uşor de însuşit şi care pot fi folosite şi de persoane cu puţine cunoştinţe de matematică. O astfel de metodă este cunoscută astăzi de foarte mulţi economişti sub denumirea de metodă simplex pentru rezolvarea problemei de programare liniară.

Metoda simplex se bazează însă pe o metodă clasică, mai puţin cunoscută, numită metoda eliminării a lui Jordan. Întrucît metoda este foarte

Scanned with OKEN Scanner

simplă cu multe aplicaţii şi cu toate acestea puţin cunoscută, intenţionăm să o prezentăm aici împreună cu cîteva dintre aplicaţiille ei mai importante.

În esenţă metoda eliminării a lui Jordan se referă la eliminarea unor variabile independente dintr-un sistem de forme liniare date.

Astfel, să considerăm un sistem format din m forme liniare:

$$
\begin{aligned}
y_1 &= a_{11}x_1 + a_{12}x_2 + \ldots + a_{1s}x_s + \ldots + a_{1n}x_n \\
y_2 &= a_{21}x_1 + a_{22}x_2 + \ldots + a_{2s}x_s + \ldots + a_{2n}x_n \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
y_r &= a_{r1}x_1 + a_{r2}x_2 + \ldots + a_{rs}x_s + \ldots + a_{rn}x_n \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
y_m &= a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{ms}x_s + \ldots + a_{mn}x_n
\end{aligned}
\tag{4.1}
$$

care se poate scrie matricial sub forma:

$$\mathbf{y} = \mathbf{Ax}.$$

Ne propunem să eliminăm una din variabilele independente, adică s-o exprimăm dintr-una din ecuaţii şi apoi s-o înlocuim în restul formelor liniare (4.1).

Să presupunem că eliminăm variabila independentă x_s din forma liniară y_r. Evident că pentru aceasta este necesar ca x_s să figureze în forma y_r, adică $a_{rs} \neq 0$.

Exprimîndu-l pe x_s din forma y_r avem:

$$x_s = \frac{1}{a_{rs}}\left[-\sum_{k=1}^{n}{}^{(s)}\, a_{rk}x_k + y_r\right], \tag{4.2}$$

unde prin $\Sigma^{(s)}$ se înţelege însumarea după indicele k de la 1 la n cu excepţia lui $k = s$.

Înlocuind x_s din (4.2) în forma y_i, avem

$$y_i = \sum_{k=1}^{n} a_{ik}x_k = \sum_{k=1}^{n}{}^{(s)}\, a_{ik}x_k + a_{is}x_s = \sum_{k=1}^{n}{}^{(s)}\, a_{ik}x_k - \frac{a_{is}}{a_{rs}}\sum_{k=1}^{n}{}^{(s)}\, a_{rk}x_k + \frac{a_{is}}{a_{rs}}y_r =$$

$$= \frac{1}{a_{rs}}\left[\sum_{k=1}^{n}{}^{(s)}\,(a_{ik}a_{rs} - a_{is}a_{rk})x_k + a_{is}y_r\right].$$

Astfel, notînd cu

$$b_{ik} = a_{ik}a_{rs} - a_{is}a_{rk}, \qquad (i \neq r, \quad k \neq s.) \tag{4.3}$$

Scanned with OKEN Scanner

avem

$$y_i = \frac{1}{a_{rs}}\left[\sum_{k=1}^{n}{}^{(s)} b_{ik}x_k + a_{is}y_r\right], \quad (i \neq r). \tag{4.4}$$

Pentru a efectua acest calcul mecanic, vom ataşa sistemului (4.4) un tabel (numit uneori tabel simplex):

(4.5)

	x_1	x_2	...	x_s	...	x_n
$y_1 =$	a_{11}	a_{12}	...	a_{1s}	...	a_{1n}
$y_2 =$	a_{21}	a_{22}	...	a_{2s}	...	a_{2n}
...	...	...	...	...	...	...
$y_r =$	a_{r1}	a_{r2}	...	$\boxed{a_{rs}}$	...	a_{rn}
...	...	...	...	...	...	...
$y_m =$	a_{m1}	a_{m2}	...	a_{ms}	...	a_{mn}

După efectuarea eliminării lui x_s din forma y_r, tabelul (4.5) se va transforma, ţinînd seama de (4.2) şi (4.4), în tabelul:

(4.6)

	x_1	x_2	...	y_r	...	x_n	
$y_1 =$	b_{11}	b_{12}	...	a_{1s}	...	b_{1n}	
$y_2 =$	b_{21}	b_{22}	...	a_{2s}	...	b_{2n}	
...	...	...	...	...	...	...	
$x_s =$	$-a_{r1}$	$-a_{r2}$	...	1	...	$-a_{rn}$	$: a_{rs}$
...	...	...	...	...	...	...	
$y_m =$	b_{m1}	b_{m2}	...	a_{ms}	...	b_{mn}	

unde b_{ik} sînt calculaţi după formula (4.3) şi toate elementele din tabel urmează să fie apoi împărţite cu a_{rs}.

Trecerea de la tabelul (4.5) la tabelul (4.6) se va numi, după STIEFEL E. [69] *un pas Jordan*, iar coeficientul a_{rs} — *coeficient pivot*. Linia şi coloana coeficientului pivot se vor numi respectiv *linia pivot şi coloana pivot*.

Analizînd tabelele (4.5) şi (4.6) se vede că după ce în tabelul (4.5) s-a ales un element pivot a_{rs} (care de obicei se încadrează într-un pătrăţel) diferit de zero, un pas Jordan se efectuează după următoarele cinci reguli:

R.1. Elementul pivot se înlocuieşte cu 1;

R.2. Celelalte elemente din coloana pivot rămîn neschimbate.

Scanned with OKEN Scanner

R.3. Celelalte elemente din linia pivot își schimbă semnul;

R.4. Elementele din restul liniilor și coloanelor se calculează după formula (4.3).

R.5. Toate elementele noului tabel se împart cu elementul pivot.

Observația 4.1. Formula (4.3) se va numi regula determinantului de ordinul doi, deoarece se vede că ea se poate scrie sub forma

$$\begin{vmatrix} a_{ik} & a_{is} \\ a_{rk} & a_{rs} \end{vmatrix}$$

sau

$$-\begin{vmatrix} a_{is} & a_{ik} \\ a_{rs} & a_{rk} \end{vmatrix}$$

adică este determinantul de ordinul doi în care prima diagonală este cea care conține elementul pivot.

Evident că se poate proceda la fel cu tabelul (4.6), adică se poate efectua un nou pas Jordan, dacă există cel puțin un element pivot diferit de zero. Întrebarea este: cîți pași Jordan se pot efectua pornind de la Tabelul inițial? Răspunsul este dat de Teorema lui *Steinitz*, care afirmă că numărul pașilor Jordan este egal cu numărul formelor (4.1) liniar independente. De altfel, această teoremă este evidentă aici, întrucît efectuarea unui nou pas Jordan este imposibilă atunci și numai atunci cînd se ajunge la un tabel de forma:

(4.7)

	y_1	$\dots$	y_k	x_{k+1}	$\dots$	x_n
$x_1 =$	c_{11}	$\dots$	c_{1k}	$c_{1,k+1}$	$\dots$	c_{1n}
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$x_k =$	c_{k1}	$\dots$	c_{kk}	$c_{k,k+1}$	$\dots$	c_{kn}
$y_{k+1} =$	$c_{k+1,1}$	$\dots$	$c_{k+1,k}$	0	$\dots$	0
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$y_m =$	c_{m1}	$\dots$	c_{mk}	0	$\dots$	0

Din acest tabel se vede că funcțiile $y_{k+1}, \dots, y_n$ sînt liniar dependente de $y_1, y_2, \dots, y_k$.

Vom da acum cîteva aplicații, dintre cele mai importante, ale metodei eliminării a lui Jordan.

1. *Inversarea matricilor.* Să presupunem că în sistemul (4.1) $m = n$ să fie $\mathbf{A} = (a_{ik})$ matricea sistemului (4.1) presupusă nesingulară. Atunci rezultă că toate cele n forme liniare sînt liniar independente și de aceea

Scanned with OKEN Scanner

se pot efectua n paşi Jordan după care Tabelul (4.5) se va transforma într-un tabel de forma:

$$\begin{array}{c|cccc} & y_1 & y_2 & \dots & y_n \\ \hline x_1 = & c_{11} & c_{12} & \dots & c_{1n} \\ \dots & c_{21} & c_{22} & \dots & c_{2n} \\ x_n = & c_{n1} & c_{n2} & \dots & c_{nn} \end{array}$$

de unde rezultă evident: $\mathbf{A}^{-1} = (c_{ik})$.

2. *Rangul unei matrici.* Metoda eliminării a lui Jordan este utilă şi pentru calculul rangului unei matrici. Astfel să presupunem că avem matricea

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

pentru a calcula rangul matricei $\mathbf{A}$ e suficient să construim tabelul:

$$\begin{array}{c|ccc} & x_1 & \dots & x_n \\ \hline y_1 = & a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots & \dots \\ y_m = & a_{m1} & \dots & a_{mn} \end{array}$$

şi să efectuăm *toţi paşii Jordan posibili.* Rangul matricei $\mathbf{A}$ va fi egal cu numărul paşilor Jordan efectuaţi. Mai observăm că din Tabelul (4.7), la care se va ajunge după k paşi (k fiind egal cu rangul matricei $\mathbf{A}$), se pot citi şi coeficienţii dependenţelor liniare a celorlalte $m - k$ linii de primele k linii.

3. *Sisteme de n ecuaţii cu n necunoscute.* Să considerăm sistemul de ecuaţii liniare

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n = a_i, \qquad (i = 1, 2, \dots, n)$$

în care rang $\mathbf{A}$ = rang $(a_{ik}) = n$.

Pentru rezolvarea sistemului dat se pot folosi două variante ale metodei eliminării a lui Jordan. Astfel, în prima variantă, se scrie sistemul sub forma tabelului

$$\begin{array}{c|cccc} & x_1 & x_2 & \dots & x_n \\ \hline a_1 = & a_{11} & a_{12} & \dots & a_{1n} \\ \dots & \dots & \dots & \dots & \dots \\ a_n = & a_{n1} & a_{n2} & \dots & a_{nn} \end{array}$$

Scanned with OKEN Scanner

şi se efectuează cei n paşi Jordan posibili, după care se va ajunge la tabelul

	a_1	a_2	...	a_n
$x_1 =$				
. .		$\mathbf{A}^{-1}$		
$x_n =$				

în care se citeşte soluţia.

În a doua variantă sistemul se scrie sub forma tabelului următor:

	x_1	x_2	...	x_n	1
$0 =$	a_{11}	a_{12}	...	a_{1n}	$-a_1$
. .	.	.	.	.	. .
$0 =$	a_{n1}	a_{n2}	...	a_{nn}	$-a_n$

Această formă de scriere este mai comodă, deoarece după fiecare pas Jordan în noul tabel se poate omite coloana pivot şi astfel de la pas la pas tabelul devine din ce în ce mai simplu. Menţionăm că aici coloana pivot nu poate figura decît printre primele n coloane.

Efectuînd n paşi Jordan vom ajunge la tabelul

	1
$x_1 =$	b_1
. .	. .
$x_n =$	b_n

Observăm că metoda cunoscută a lui Gauss se deosebeşte de aceasta numai prin aceea că după fiecare pas se omite nu numai coloana pivot ci şi linia pivot, scriind separat expresia lui x_i din linia pivot considerată. Astfel după ultimul pas Jordan tabelul se reduce la o singură celulă care va da valoarea ultimei variabile. Celelalte valori se calculează din aproape în aproape începînd cu penultima, cu ajutorul valorilor deja obţinute folosind expresiile necunoscutelor scrise separat.

4. *Sisteme generale de ecuaţii liniare.* Să considerăm acum sistemul de ecuaţii

$$y_i = a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n - a_i = 0, \qquad (i = 1, 2, \ldots, m).$$

în raport cu necunoscutele: $x_1, x_2, \ldots, x_n$.

Scanned with OKEN Scanner

Să presupunem că rangul matricei $\mathbf{A} = (a_{ik})$ este egal cu r. Atunci, pornind de la tabelul

$$\begin{array}{c|cccc|c} & x_1 & x_2 & \dots & x_n & 1 \\ \hline y_1 = & a_{11} & a_{12} & \dots & a_{1n} & -a_1 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ y_m = & a_{m1} & a_{m2} & \dots & a_{mn} & -a_m \end{array}$$

după r paşi Jordan şi după o eventuală renumerotare, se ajunge la un tabel de forma

$$\begin{array}{c|cccccc|c} & y_1 & \dots & y_r & x_{r+1} & \dots & x_n & 1 \\ \hline x_1 = & c_{11} & \dots & c_{1r} & c_{1,r+1} & \dots & c_{1n} & c_1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ x_r = & c_{r1} & \dots & c_{rr} & c_{r,r+1} & \dots & c_{rn} & c_r \\ y_{r+1} = & c_{r+1,1} & \dots & c_{r+1,r} & 0 & \dots & 0 & c_{r+1} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ y_m = & c_{m1} & \dots & c_{mr} & 0 & \dots & 0 & c_m \end{array} \tag{4.8}$$

Sistemul considerat este compatibil dacă şi numai dacă există un sistem de numere $x_1, x_2, \dots, x_n$ pentru care toţi $y_i = 0$, $i = 1, 2, \dots, n$. Din tabelul (4.8) se vede însă că acest lucru se realizează atunci şi numai atunci cînd $c_{r+1} = c_{r+2} = \dots = c_m = 0$. În acest caz Tabelul (4.8) dă şi soluţia generală a sistemului, făcînd $y_1 = \dots = y_r = 0$.

Făcînd comparaţie cu metoda greoaie care se foloseşte la studiul sistemelor liniare, bazată pe teorema lui Kronecker-Capelli, care presupune mai întîi calculul rangului a două matrici şi apoi rezolvarea unui sistem, avantajul acestei metode este evident.

Cea mai importantă aplicaţie a metodei expuse o constituie metoda simplex, folosită în problema de programare liniară, care va fi tratată la punctul următor.

§ 5. Metoda simplex

Deoarece în mai toate cărţile de programare liniară, metoda simplex descrisă se aplică după ce s-au transformat inegalităţile în egalităţi după introducerea variabilelor auxiliare, considerăm necesar să dăm aici o altă variantă a metodei simplex, după părerea noastră mai simplă, dată de ZUHOVIŢKI S. I. şi AVDEEVA L. I. [86], care nu necesită transformarea sistemului de inegalităţi în egalităţi şi care se bazează pe metoda eliminării a lui Jordan descrisă mai sus. Pe această variantă a metodei simplex se vor baza şi unele metode de rezolvare a problemelor de programare care vor fi date în paragrafele următoare.

Scanned with OKEN Scanner

Pentru aceasta vom descrie mai întîi *metoda modificată a eliminării a lui Jordan*. Astfel, să considerăm tabelul

$$
\begin{array}{c|cccccc}
 & (t_1) & (t_2) & & (t_s) & & (t_n) \\
 & -x_1 & -x_2 & & -x_s & & -x_n \\
\hline
y_1 = & a_{11} & a_{12} & \dots & a_{1s} & \dots & a_{1n} \\
\dots & \dots & \dots & & \dots & & \dots \\
y_r = & a_{r1} & a_{r2} & \dots & a_{rs} & \dots & a_{rn} \\
\dots & \dots & \dots & & \dots & & \dots \\
y_m = & a_{m1} & a_{m2} & \dots & a_{ms} & \dots & a_{mn}
\end{array} \tag{5.1}
$$

Alegînd ca element pivot $a_{rs} \neq 0$ şi efectuînd un pas Jordan obişnuit în raport cu variabilele t_k, vom ajunge la tabelul:

$$
\begin{array}{c|cccccc}
 & (t_1) & (t_2) & & & & (t_n) \\
 & -x_1 & -x_2 & \dots & y_r & \dots & -x_n \\
\hline
y_1 = & b_{11} & b_{12} & \dots & a_{1s} & \dots & b_{1n} \\
\dots & \dots & \dots & & \dots & & \dots \\
(t_s) = -x_s = & -a_{r1} & -a_{r2} & \dots & 1 & \dots & -a_{rn} \\
\dots & \dots & \dots & & \dots & & \dots \\
y_m = & b_{m1} & b_{m2} & \dots & a_{ms} & \dots & b_{mn}
\end{array} : a_{rs} \tag{5.2}
$$

Dacă omogenizăm Tabelul (5.2), înmulţind cu -1 coloana şi linia pivot, obţinem tabelul

$$
\begin{array}{c|cccccc}
 & -x_1 & -x_2 & & -y_r & & -x_n \\
\hline
y_1 = & b_{11} & b_{12} & \dots & -a_{1s} & \dots & b_{1n} \\
\dots & \dots & \dots & & \dots & & \dots \\
x_1 = & a_{r1} & a_{r2} & \dots & 1 & \dots & a_{rn} \\
\dots & \dots & \dots & & \dots & & \dots \\
y_m = & b_{m1} & b_{m2} & \dots & -a_{ms} & \dots & b_{mn}
\end{array} : a_{rs}
$$

Deci reţinem că un pas Jordan modificat se efectuează după aceleaşi reguli ca şi cel obişnuit, cu singura modificare că aici linia pivot nu îşi schimbă semnul, iar coloana pivot se scrie cu semn schimbat. De fapt pe această metodă se va baza metoda simplex pe care intenţionăm s-o descriem.

Să considerăm acum următoarea problemă generală de programare liniară:

$$\max z = \max \left[p_1 x_1 + p_2 x_2 + \dots + p_n x_n\right] \tag{5.3}$$

$$a_{i1} x_1 + a_{i2} x_2 + \dots + a_{in} x_n \leqslant b_i, \qquad (i = 1, 2, \dots, m). \tag{5.4}$$

Scanned with OKEN Scanner

Vom scrie condiţiile (4) sub forma:

$$(5.5) \qquad y_i \equiv a_{i1}(-x_1) + a_{i2}(-x_2) + \ldots + a_{in}(-x_n) + b_i \geqslant 0, \quad (i = 1, 2, \ldots, m).$$

Metoda simplex pentru această problemă generală se va descompune în trei etape:

1°. Eliminarea variabilelor independente.

2°. Găsirea unei soluţii admisibile (o soluţie a sistemului de inegalităţi (5.4)).

3°. Îmbunătăţirea soluţiilor admisibile şi determinarea soluţiei optime.

1°. Vom începe cu prima etapă. Pentru aceasta ataşăm problemei (5.3) — (5.4) tabelul simplex:

(5.6)

	$-x_1$	$-x_2$	$\ldots$	$-x_n$	1
$y_1 =$	a_{11}	a_{12}	$\ldots$	a_{1n}	b_1
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$y_m =$	a_{m1}	a_{m2}	$\ldots$	a_{mn}	b_m
$z =$	$-p_1$	$-p_2$	$\ldots$	$-p_n$	0

Putem presupune că rang$\mathbf{A} = n$, căci în caz contrar problema de programare liniară se reduce la o problemă de minim cu legături.

Atunci, pornind de la Tabelul (5.6) vom putea efectua n paşi Jordan modificaţi după care vom ajunge, după o eventuală renumerotare a variabilelor, la tabelul:

(5.7)

	$-y_1$	$-y_2$	$\ldots$	$-y_n$	1
$x_1 =$	c_{11}	c_{12}	$\ldots$	c_{1n}	c_1
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$x_n =$	c_{n1}	c_{n2}	$\ldots$	c_{nn}	c_n
$y_{n+1} =$	$c_{n+1,1}$	$c_{n+1,2}$	$\ldots$	$c_{n+1,n}$	c_{n+1}
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$y_m =$	c_{m1}	c_{m2}	$\ldots$	c_{mn}	c_m
$z =$	q_1	q_2	$\ldots$	q_n	Q

Întrucît expresiile variabilelor independente $x_1, x_2, \ldots, x_n$ din (5.7) intervin numai după ce se găseşte soluţia optimă, ele pot fi omise din Tabelul (5.7), exprimîndu-le sub forma:

$$(5.8) \qquad x_k = -\sum_{j=1}^{n} c_{kj}\, y_j + c_k \quad , \qquad (k = 1, 2, \ldots, n).$$

Scanned with OKEN Scanner

Atunci Tabelul (5.7) devine

(5.9)

		$-y_1$	$-y_2$		$-y_n$	1
y_{n+1}	$=$	$c_{n+1,1}$	$c_{n+1,2}$	$\ldots$	$c_{n+1,n}$	c_{n+1}
$\ldots$		$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
y_m	$=$	c_{m1}	c_{m2}	$\ldots$	c_{mn}	c_m
z	$=$	q_1	q_2	$\ldots$	q_n	Q

şi cu aceasta prima etapă se încheie.

Observaţia 5.1. Tabelul (5.9) corespunde următoarei probleme de programare liniară

$$\max\,(-q_1y_1 - q_2y_2 \ldots . -q_n y_n + Q)$$

în condiţiile:

$$y_i = -c_{i1}y_1 - c_{i2}y_2 - \ldots - c_{in}y_n + c_i \geqslant 0, \quad (i = n+1, \ldots, m)$$

$$y_1 \geqslant 0,\ y_2 \geqslant 0,\ \ldots,\ y_n \geqslant 0.$$

Aceasta se deosebeşte de cea iniţială prin faptul că aici variabilele independente sînt supuse restricţiei de nenegativitate. O asemenea problemă se mai numeşte *formă canonică a problemei de programare liniară.*

Astfel, dacă problema iniţială este dată sub formă canonică, adică este de forma:

$$\max\,\{\mathbf{cx} + \alpha \mid \mathbf{Ax} \leqslant \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}\},$$

atunci etapa întîi de eliminare a variabilelor independente în algoritmul simplex nu este necesară.

2°. Pentru a găsi o soluţie admisibilă a sistemului (5.4) (sau (5.5)), se porneşte de la Tabelul (5.9) şi se efectuează un şir de paşi Jordan modificaţi cu o alegere convenabilă a elementului pivot.

Algoritmul constă în următoarele:

1) Dacă $c_{n+1} \geqslant 0,\ c_{n+2} \geqslant 0,\ \ldots,\ c_m \geqslant 0,$ atunci evident o soluţie admisibilă este $y_1 = y_2 = \ldots = y_n = 0,\ y_i = c_i,\ i = n+1, \ldots, m.$

2) Dacă există $c_r < 0, \quad n+1 \leqslant r \leqslant m$ şi

$$c_{r1} \geqslant 0, \quad c_{r2} \geqslant 0, \ \ldots,\ c_{rn} \geqslant 0,$$

atunci sistemul (5.4) este incompatibil şi problema (5.3) — (5.4) nu are sens.

Într-adevăr, în acest caz

$$y_r = -\sum_{k=1}^{n} c_{rk}y_k + c_r < 0$$

oricare ar fi $y_1 \geqslant 0,\ y_2 \geqslant 0,\ \ldots,\ y_n \geqslant 0.$

Scanned with OKEN Scanner

3) Dacă există $c_r < 0$, $n+1 \leqslant r \leqslant m$ şi există $c_{rs} < 0$, atunci coloana de indice s este coloana pivot în care se alege elementul pivot $c_{i_0 s}$ astfel ca

$$0 \leqslant \frac{c_{i_0}}{c_{i_0 s}} = \min_{n+1 \leqslant i \leqslant m} \frac{c_i}{c_{is}}.$$

Observaţia 5.2. În cazul cînd

$$\min_{n+1 \leqslant i \leqslant m} \frac{c_i}{c_{is}} = \frac{c_{i_0}}{c_{i_0 s}} = 0$$

elementul $c_{i_0 s}$ se va lua ca element pivot numai dacă $c_{i_0 s} > 0$. În caz contrar se trece la raportul următor sau la o altă coloană pivot (dacă o asemenea coloană există).

Cu elementul pivot astfel determinat se efectuează un nou pas Jordan modificat.

Într-adevăr, dacă $c_r < 0$, $n+1 \leqslant r \leqslant m$, atunci rezultă că hiperplanul

$$y_r = 0$$

separă vîrful $y_1 = y_2 = \ldots = y_n = 0$ de domeniul soluţiilor admisibile:

$$\Omega = \{\mathbf{x} \in \mathbf{R}^n | \mathbf{A}\mathbf{x} \leqslant \mathbf{b}\},$$

căci în acest vîrf $y_r = c_r < 0$.

Punînd $y_s = t > 0$, $y_i = 0$, $i \neq s$, $i = 1, 2, \ldots, n$, avem

$$y_r = -c_{rs} t + c_r.$$

Pentru ca, deplasîndu-se pe o muchie ce pleacă din vîrful $y_1 = y_2 = \ldots = y_n = 0$, să ne apropiem de planul separator $y_r = 0$ este necesar şi suficient ca $|y_r|$ să descrească, adică pentru $t > 0$ suficient de mic să avem

$$|-c_{rs} t + c_r| < |c_r|. \tag{5.10}$$

Deoarece însă $c_r < 0$, rezultă că pentru t suficient de mic $-c_{rs} t + c_r < 0$ şi deci (5.10) devine

$$-(-c_{rs} t + c_r) < -c_r,$$

de unde

$$c_{rs} t < 0$$

Scanned with OKEN Scanner

Prin urmare $c_{rs} < 0$. Să presupunem că există $c_{rs} < 0$. Atunci ne vom deplasa pe această muchie pînă ce întîlnim un nou hiperplan $y_i = 0$, $i = n + 1, \ldots, m$. Deoarece însă

$$y_i = -c_{is}t + c_i = 0,$$

rezultă

$$0 \leqslant t = \frac{c_i}{c_{is}},$$

de unde obţinem criteriul de alegere a elementului pivot.

Cu excepţia cunoscutului caz de degenerare pe care nu-l discutăm aici, după un număr finit de paşi se va ajunge la o soluţie admisibilă sau la constatarea că sistemul (5.4) este incompatibil, deoarece deplasarea se face din vîrf în vîrf (care sînt în număr finit) şi algoritmul este strict monoton.

3°. Să presupunem că am găsit o soluţie admisibilă a sistemului (5.4), adică în tabelul (5.9)

$$c_{n+1} \geqslant 0, \; c_{n+2} \geqslant 0, \; \ldots, \; c_m \geqslant 0.$$

Atunci analizăm ultima linie a acestui tabel.

1) Dacă $q_1 \geqslant 0, \; q_2 \geqslant 0, \; \ldots, \; q_n \geqslant 0,$
atunci am ajuns la soluţia optimă, căci evident

$$-\sum_{k=1}^{n} q_k y_k + Q \leqslant Q$$

oricare ar fi $y_1 \geqslant 0, \; y_2 \geqslant 0, \; \ldots, \; y_n \geqslant 0$. Maximul funcţiei de scop este Q şi este atins în vîrful $y_1 = y_2 = \ldots = y_n = 0$. Coordonatele vîrfului se citesc din (5.8).

2) Dacă există $q_s < 0$ şi

$$c_{is} \leqslant 0, \quad \forall i \in \{n + 1, \ldots, m\}$$

atunci problema (5.3) — (5.4) nu are soluţie căci

$$\sup_{x \in \Omega} z(\mathbf{x}) = +\infty$$

Într-adevăr, în acest caz, punînd

$$y_s = t > 0, \; y_k = 0, \; k \neq s, \; k = 1, 2, \ldots, n$$

Scanned with OKEN Scanner

avem

$$y_i = -c_{is}t + c_i \geqslant 0, \qquad \forall i \in \{n+1, \ldots, m\}$$

oricare ar fi $t > 0$. Pe de altă parte avem

$$z = -q_s t + Q \to +\infty, \qquad t \to +\infty.$$

3) Dacă $q_s < 0$ şi există $c_{is} > 0$, $n+1 \leqslant i \leqslant m$, atunci coloana de indice s este coloana pivot, din care se alege elementul pivot după aceeaşi regulă ca la etapa a doua. Cu elementul pivot astfel determinat se efectuează un nou pas Jordan.

Într-adevăr, cînd coloana de indice s este coloana pivot, atunci relaţiile

$$y_i = -c_{is}t + c_i = 0, \qquad (i = n+1, \ldots, m),$$

de unde

$$t = \frac{c_i}{c_{is}}, \tag{5.11}$$

ne conduc la alegerea elementului pivot, ca fiind numitorul celui mai mic raport (5.11) pozitiv (cel puţin un asemenea raport există dacă există cel puţin un coeficient $c_{is} > 0$). Fie

$$0 < \frac{c_{i_0}}{c_{i_0 s}} \leqslant \frac{c_i}{c_{is}}, \qquad n+1 \leqslant i \leqslant m.$$

Atunci pentru $t_0 = \frac{c_{i_0}}{c_{i_0 s}}$, avem

$$z(t_0) = -\frac{c_{i_0}}{c_{i_0 s}} \cdot q_s + Q > Q,$$

ceea ce arată că funcţia de scop creşte. Datorită acestui fapt şi datorită existenţei unui număr finit de vîrfuri ale domeniului Ω, rezultă că după un număr finit de paşi se va ajunge sau la soluţia optimă căutată, sau la constatarea că maximul funcţiei de scop pe Ω este $+\infty$.

Scanned with OKEN Scanner

Pentru ilustrare, considerăm următorul exemplu. Să se maximizeze funcţia

$$z = -3x_1 + 6x_2$$

în condiţiile

$$\begin{aligned} y_1 &= x_1 + 2x_2 + 1 \geqslant 0 \\ y_2 &= 2x_1 + x_2 - 4 \geqslant 0 \\ y_3 &= x_1 - x_2 + 1 \geqslant 0 \\ y_4 &= x_1 - 4x_2 + 13 \geqslant 0 \\ y_5 &= -4x_1 + x_2 + 23 \geqslant 0. \end{aligned}$$

Tabelul simplex corespunzător acestei probleme este:

	$-x_1$	$-x_2$	1
$y_1 =$	$\boxed{-1}$	-2	1
$y_2 =$	-2	-1	-4
$y_3 =$	-1	1	1
$y_4 =$	-1	4	13
$y_5 =$	4	-1	23
$z =$	3	-6	0

1°. Vom începe cu eliminarea variabilelor independente. După un pas Jordan modificat cu elementul pivot -1 din linia şi coloana întîi, ajungem la tabelul:

	$-y_1$	$-x_2$	1
$x_1 =$	-1	2	-1
$y_2 =$	-2	$\boxed{3}$	-6
$y_3 =$	-1	3	0
$y_4 =$	-1	6	12
$y_5 =$	4	-9	27
$z =$	3	-12	3

Scanned with OKEN Scanner

Al doilea pas Jordan modificat se va efectua cu elementul pivot situat în linia și coloana a doua, după care se ajunge la tabelul

	$-y_1$	$-y_2$	1
$x_1 =$	$\frac{1}{3}$	$-\frac{2}{3}$	3
$x_2 =$	$-\frac{2}{3}$	$\frac{1}{3}$	-2
$y_3 =$	1	-1	6
$y_4 =$	3	-2	24
$y_5 =$	-2	3	9
$z =$	-5	4	-21

Scriind separat expresiile lui x_1 și x_2:

$$x_1 = -\frac{1}{3}y_1 + \frac{2}{3}y_2 + 3 \tag{5.12}$$

$$x_2 = \frac{2}{3}y_1 - \frac{1}{3}y_2 - 2$$

tabelul devine

	$-y_1$	$-y_2$	1
$y_3 =$	$\boxed{1}$	-1	6
$y_4 =$	3	-2	24
$y_5 =$	-2	3	9
$z =$	-5	4	-21

2°. Deoarece coloana termenilor liberi în acest tabel este pozitivă, rezultă că etapa a doua s-a încheiat, $y_1 = y_2 = 0$ fiind o soluție admisibilă.

3°. Deoarece un element în ultima linie este negativ și pe prima coloană există elemente pozitive, scriem rapoartele:

$$\frac{6}{1}, \quad \frac{24}{3}, \quad \frac{9}{-2}.$$

Scanned with OKEN Scanner

Raportul pozitiv minim este $\frac{6}{1}$, deci 1 este elementul pivot, cu care vom efectua un nou pas Jordan modificat. În urma acestui pas Jordan obţinem tabelul

	$-y_3$	$-y_2$	1
$y_1 =$	1	−1	6
$y_4 =$	−3	**1**	6
$y_5 =$	2	1	21
$z =$	5	−1	9

Avem şi de data aceasta în ultima linie un element negativ şi vom efectua un nou pas Jordan modificat, luînd ca element pivot pe 1 din linia şi coloana a doua. După efectuarea acestui pas Jordan obţinem tabelul

	$-y_3$	$-y_4$	1
$y_1 =$	−2	1	12
$y_2 =$	−3	1	6
$y_5 =$	5	−1	15
$=$	2	1	15

Întrucît elementele ultimei linii sînt pozitive, rezultă că algoritmul s-a încheiat şi avem

$$\max z(\mathbf{x}) = 15$$

şi este atins pentru $y_3 = y_4 = 0$. Deoarece $y_1 = 12$, $y_2 = 6$, din (5.12) obţinem:

$$x_1 = 3, \qquad x_2 = 4.$$

Prin urmare

$$\max z(x_1, x_2) = z(3, 4) = 15.$$

Scanned with OKEN Scanner

CAPITOLUL II

METODE GENERALE DE REZOLVARE A PROBLEMELOR DE PROGRAMARE NELINIARĂ

§ 6. Metoda planelor secante (Cutting—plane Method)

Metoda descrisă în § 3. Capitolul I, bazată pe determinarea traiectoriilor ortogonale este o metodă mai mult teoretică, fără o importanţă practică prea mare. Greutatea constă evident în integrarea sistemului de ecuaţii diferenţiale care determină traiectoriile ortogonale.

De aceea s-a căutat să se liniarizeze problema de programare neliniară. Natural că metoda liniarizării conduce uneori la probleme de programare liniară cu un volum de calcule foarte mare, care rezultă din numărul foarte mare de inegalităţi la care se ajunge. Cu toate acestea problema poate fi rezolvată cu calculatoare, ţinînd seama de faptul amintit deja, că pentru metoda simplex există programe bine puse la punct şi verificate în practică, care permit rezolvarea în condiţii bune a problemei de programare liniară.

În ultimii ani au apărut o serie de metode bazate pe procedeul liniarizării. Printre acestea se numără şi *metoda planelor secante*. Această metodă este prezentată în trei variante. Prima variantă a fost dată de CHENEY, E. W. şi GOLSTEIN, A. A. [13] într-o lucrare intitulată: „Newton's method for convex programming and Tchebycheff approximation", apărută în 1959. A doua variantă a fost dată de KELLEY, J. E. [35] intitulată „The cutting — plane method for solving convex programs", apărută în anul 1960. A treia variantă: a fost dată de HARTLEY, H. O. şi HOCKING, R. R. în lucrarea intitulată: „Convex programming by tangential approximation" [26], apărută în anul 1963.

În cele ce urmează vom prezenta varianta a doua. De altfel toate cele trei variante sînt foarte asemănătoare. Pentru demonstraţii se poate consulta, de exemplu, cartea lui COLLATZ L., şi WETTERLING, W. [14], care conţine această metodă.

Metoda planelor secante se referă la următoarea problemă: *să se găsească un punct* $\mathbf{x} \in \mathbf{R}^n$, *astfel ca*

$$\min f(\mathbf{x}) = \min \mathbf{cx}, \tag{6.1}$$

cînd punctele $\mathbf{x}$ *verifică restricţiile*

$$g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots, m), \tag{6.2}$$

Scanned with OKEN Scanner

unde g_j, $j = 1,2, \ldots, m$, sînt convexe, iar

$$\mathbf{c} = (c_1, c_2, \ldots, c_n), \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}.$$

Prin urmare, în această problemă funcţia de scop se presupune liniară, iar restricţiile în general neliniare.

De altfel orice problemă de programare convexă se poate reduce la o problemă de tipul (6.1) — (6.2). Într-adevăr, dacă funcţia de scop f nu este liniară însă este o funcţie convexă, atunci se consideră următoarea problemă de programare de tipul amintit: *să se minimizeze funcţia liniară*

$$\bar{f}(x_1, \ldots, x_n, x_{n+1}) = x_{n+1}$$

în condiţiile (6.2) *la care se adaugă restricţia*

$$f(x_1, x_2, \ldots, x_n) - x_{n+1} \leqslant 0.$$

Se vede imediat că dacă $\tilde{\mathbf{x}} \in \mathbf{R}^n$ este o soluţie optimă a problemei (6.1) — (6.2), atunci punctul $\tilde{\mathbf{x}} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_{n+1})$, unde

$$\tilde{x}_{n+1} = f(\tilde{\mathbf{x}})$$

este soluţia problemei a doua şi reciproc.

Pentru ca algoritmul bazat pe metoda planelor secante să fie convergent, este necesar să fie îndeplinite următoarele două condiţii:

1) Mulţimea punctelor admisibile Ω, definită de inegalităţile (6.2) este conţinută într-un poliedru

$$P_0 = \{\mathbf{x} \in \mathbf{R} \mid \mathbf{A}\mathbf{x} \leqslant \mathbf{b}\},$$

unde P_0 este mărginit.

2) Funcţiile g_j, $j = 1, 2, \ldots, m$, sînt convexe şi diferenţiabile pe P_0, astfel încît

$$\| \nabla g_j(\mathbf{x}) \| \leqslant K, \qquad \mathbf{x} \in P_0 \ (j = 1, 2, \ldots, m).$$

Procedeul constă în rezolvarea a cîte unei probleme de programare liniară la fiecare pas, mai întîi pe poliedrul P_0 şi apoi pe rînd pe cîte un sub-

Scanned with OKEN Scanner

poliedru al acestuia P_k, $k = 0,1,\ldots$. Dacă $\mathbf{x}^k$ este soluţia problemei de programare liniară corespunzătoare poliedrului P_k şi funcţiei (6.1), atunci algoritmul constă în efectuarea succesivă a următorilor doi paşi:

1°. Se rezolvă problema de programare liniară

$$f(\mathbf{x}) = \mathbf{cx} = \min., \quad \mathbf{x} \in P_k.$$

Fie $\mathbf{x}^k$ soluţia acestei probleme.

2°. Dacă ν este un indice pentru care

$$g_\nu(\mathbf{x}^k) = \max_{(j)} g_j(\mathbf{x}^k),$$

atunci P_{k+1} este intersecţia mulţimii P_k cu mulţimea din $\mathbf{R}^n$, definită de inegalitatea liniară:

$$g_\nu(\mathbf{x}^k) + \nabla g_\nu(\mathbf{x}^k)\,(\mathbf{x} - \mathbf{x}^k) \leqslant 0,$$

adică

$$\nabla g_\nu(\mathbf{x}^k)\mathbf{x} \leqslant \nabla g_\nu(\mathbf{x}^k)\mathbf{x}^k - g_\nu(\mathbf{x}^k). \tag{6.3}$$

Din condiţia 2° rezultă că poliedrul P_{k+1} ia naştere din P_k printr-o secţiune plană. P_{k+1} este o parte a poliedrului P_k obţinută prin intersecţia lui P_k cu un hiperplan. De aici şi denumirea de metodă a planelor secante.

Cu acest poliedru redus se procedează apoi la fel, adică se rezolvă problema de programare liniară pentru funcţia $f(\mathbf{x})$ şi $\mathbf{x}^k \in P_{k+1}$ şi procedeul se continuă.

Se poate arăta [14] că în condiţiile 1) şi 2) acest procedeu este totdeauna convergent către soluţia optimă a problemei (6.1) — (6.2).

În cazul bidimensional metoda este ilustrată în Figura 6.1.

Aici s-a luat drept poliedru iniţial pătratul P_0. Punctul $\mathbf{x}^0$ este soluţia primei probleme de programare liniară pentru $\mathbf{x}^0 \in P_0$.

Fie

$$g_1(\mathbf{x}^0) = \max_{1,2} g_j(\mathbf{x}^0).$$

Să considerăm apoi ecuaţia planului tangent la suprafaţa

$$z = g_1(\mathbf{x})$$

în punctul $\mathbf{x} = \mathbf{x}^0$. Ecuaţia intersecţiei acestuia cu planul $z = 0$, este

$$\nabla g_1(\mathbf{x}^0)\mathbf{x} = \nabla g_1(\mathbf{x}^0)\mathbf{x}^0 - g_1(\mathbf{x}^0). \tag{6.4}$$

Scanned with OKEN Scanner

Observăm că ecuația intersecției cu planul $z = 0$ a planului tangent la suprafața $z = g_j(\mathbf{x})$ în punctul $\mathbf{x}^k$ se mai poate scrie astfel: se scrie ecuația planului tangent la suprafața $g_j(\mathbf{x}) = g_j(\mathbf{x}^k)$ care se translatează apoi de-a lungul normalei cu $-g_j(\mathbf{x}^k)$ spre domeniul soluțiilor Ω. Aici este vorba

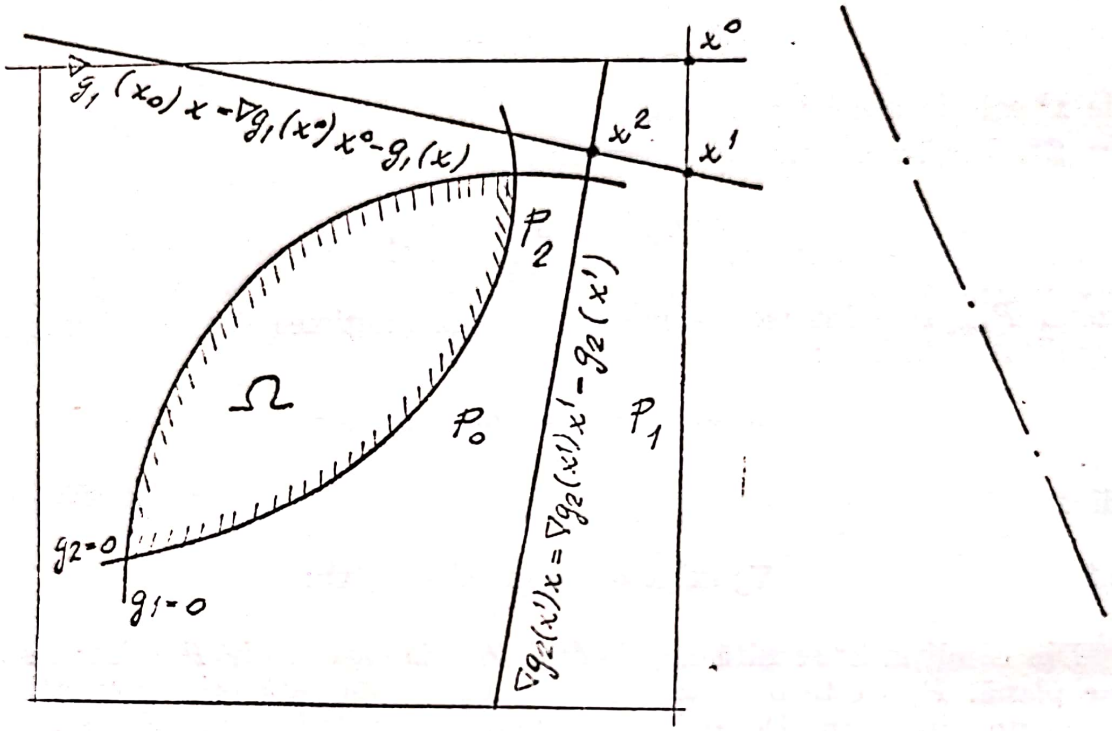

Figura 6.1.

de hiperplan tangent la o suprafață în spațiul $\mathbf{R}^n$ în timp ce în primul caz se consideră hiperplanul într-un spațiu $\mathbf{R}^{n+1}$. În exemplul nostru, în loc de ecuația planului tangent la suprafața

$$z = g_1(\mathbf{x})$$

în punctul $\mathbf{x}^0$, intersectat apoi cu planul $z = 0$, putem scrie ecuația dreptei tangente în $\mathbf{x}^0$ la curba $g_1(\mathbf{x}) = g_1(\mathbf{x}^0)$, translatată cu valoarea $-g_1(\mathbf{x}^0)$.

Dreapta (6.4) taie din pătratul P_0 o porțiune. Complementara acestei porțiuni determină poliedrul P_1. Procedînd cu P_1 la fel, adică rezolvînd problema de programare liniară:

$$f(\mathbf{x}) = \mathbf{cx} = \min., \ \mathbf{x} \in P_1,$$

vom găsi un punct $\mathbf{x}^1$, care este soluția acestei probleme. Fie

$$g_2(\mathbf{x}^1) = \max_{1,2} g_j(\mathbf{x}^1).$$

Se scrie ecuaţia planului tangent la suprafaţa $z = g_2(\mathbf{x})$ în punctul $\mathbf{x}^1$, care se intersectează apoi cu planul $z = 0$, obţinîndu-se dreapta

$$\nabla g_2(\mathbf{x}^1)\mathbf{x} = \nabla g_2(\mathbf{x}_1)\mathbf{x}^1 - g_2(\mathbf{x}^1).$$

Această dreaptă intersectată cu poligonul P_1 determină un nou poligon P_2 cu care se procedează la fel ş.a.m.d.

Soluţia problemei iniţiale va fi dată de

$$\mathbf{x}^* = \lim_{k\to\infty} \mathbf{x}^k.$$

Ca orice metodă, din punct de vedere practic, metoda descrisă are şi părţi negative şi părţi pozitive. Ca părţi pozitive amintim faptul că datorită liniarizării se poate utiliza metoda simplex. Avînd în vedere că iteraţiile sînt foarte simple, se poate întocmi un program la maşină destul de uşor. În sfîrşit ca o altă parte pozitivă a metodei, amintim faptul că aici nu trebuie să se cunoască o soluţie de pornire, întrucît aproximarea soluţiei optime se face din *„afară"*.

Ca părţi negative amintim următoarele: 1. Convergenţa nu este suficient de rapidă, în special cînd optimul se găseşte într-un vîrf; 2. O iteraţie oarecare $\mathbf{x}^k$ nu se poate utiliza ca o soluţie aproximativă întrucît nu toate iteraţiile sînt soluţii admisibile.

WOLFE, PH. [77] a arătat că în cazul cînd restricţiile sînt liniare, iar funcţia de scop este neliniară, metoda este avantajoasă.

Exemplul 6.1: Pentru ilustrare considerăm următorul exemplu

$$f(\mathbf{x}) = -x_1 \to \min \tag{6.5}$$

cînd

$$\begin{aligned} &e^{x_1} - x_2 \leqslant 0 \\ &e^{x_2} - x_3 \leqslant 0 \\ &x_3 - 10 \leqslant 0 \\ &x_1 \geqslant 0, \quad x_2 \geqslant 0, \quad x_3 \geqslant 0 \end{aligned} \tag{6.6}$$

Evident că funcţiile

$$\begin{aligned} g_1(\mathbf{x}) &= e^{x_1} - x_2 \\ g_2(\mathbf{x}) &= e^{x_2} - x_3 \\ g_3(\mathbf{x}) &= x_3 - 10 \end{aligned}$$

sînt convexe. Domeniul Ω al soluţiilor admisibile este conţinut în paralelipipedul

$$P_0 = \{\mathbf{x} \in \mathbf{R}^3 \mid 0 \leqslant x_1 \leqslant 2;\ 0 \leqslant x_2 \leqslant 3;\ 0 \leqslant x_3 \leqslant 10\}.$$

Scanned with OKEN Scanner

Astfel prima problemă de programare liniară este următoarea:

$$\min \{ -x_1 \mid \mathbf{x} \in P_0 \}. \tag{6.7}$$

Tabelul simplex corespunzător acestei este

	$-x_1$	$-x_2$	$-x_3$	1
$y_1 =$	$\boxed{1}$	0	0	2
$y_2 =$	0	1	0	3
$y_3 =$	0	0	1	10
$f =$	1	0	0	0

Efectuînd un pas Jordan modificat cu elementul pivot marcat, se ajunge la tabelul:

	$-y_1$	$-x_2$	$-x_3$	1
$x_1 =$				2
$y_2 =$				3
$y_3 =$				10
$f =$	−1	0	0	−2

care arată că soluţia problemei (6.7) este $\mathbf{x}^0 = (2, 0, 0)$ şi $f(\mathbf{x}^0) = -2$.

Avem

$$g_1(\mathbf{x}^0) = e^2,\ g_2(\mathbf{x}^0) = 1,\ g_3(\mathbf{x}^0) = -10.$$

Deci

$$\max g_j(\mathbf{x}^0) = g_1(\mathbf{x}^0) = e^2,$$

iar

$$\nabla g_1(\mathbf{x}^0) = (e^2, -1, 0),$$

Ecuaţia (6.3) devine

$$e^2 + (x_1 - 2)e^2 - x_2 \leqslant 0$$

sau

$$g_4(\mathbf{x}) = e^2 x_1 - x_2 - e^2 \leqslant 0.$$

Scanned with OKEN Scanner

Prin urmare noul poligon este

$$P_1 = P_0 \cap \{\mathbf{x} \in \mathbf{R}^3 \mid g_4(\mathbf{x}) \leqslant 0\}.$$

Pasul doi începe cu rezolvarea următoarei probleme de programare liniară

$$\min \{-x_1 \mid \mathbf{x} \in P_1\}. \tag{6.8}$$

Tabelul corespunzător acesteia este

	$-x_1$	$-x_2$	$-x_3$	1
$y_1 =$	1	0	0	2
$y_2 =$	0	1	0	3
$y_3 =$	0	0	1	10
$y_4 =$	$\boxed{e^2}$	-1	0	e^2
$f_4 =$	1	0	0	0

După doi paşi Jordan începînd cu elementul pivot marcat se ajunge la tabelul:

	$-y_4$	$-y_2$	$-x_3$	1
$y_1 =$				$1 - 3e^{-2}$
$x_2 =$				3
$y_3 =$				10
$x_1 =$				$1 + 3e^{-2}$
$f =$	$-e^{-2}$	$-e^{-2}$	0	$-1 - 3e^{-2}$

care arată că soluţia problemei (6.8) este $\mathbf{x}^1 = (1 + 3e^{-2}, 3, 0)$ şi $f(\mathbf{x}^1) = -1 - 3e^{-2} \approx -1{,}4$.

Întrucît

$$g_1(\mathbf{x}^1) \approx e^{1,4} - 3,\ g_2(\mathbf{x}^1) = e^3,\ g_3(\mathbf{x}^1) = -10,$$

rezultă că

$$\max g_j(\mathbf{x}^1) = g_2(\mathbf{x}^1) = e^3,$$

iar

$$\nabla g_2(\mathbf{x}^1) = (0,\ e^3,\ -1).$$

Scanned with OKEN Scanner

Ecuaţia (6.3) în acest caz este

$$g_5(\mathbf{x}) = e^3 x_2 - x_3 - 2e^3 \leqslant 0,$$

iar noul poligon este

$$P_2 = P_1 \cap \{\mathbf{x} \in \mathbf{R}^n \mid g_5(\mathbf{x}) \leqslant 0\}.$$

Noua problemă de programare liniară este

(6.9) $$\min \{-x_1 \mid \mathbf{x} \in P_2\}.$$

Tabelul simplex corespunzător problemei (6.9) este

	$-x_1$	$-x_2$	$-x_3$	1
$y_1 =$	1	0	0	2
$y_2 =$	0	1	0	3
$y_3 =$	0	0	1	10
$y_4 =$	$\boxed{e^2}$	-1	0	e^2
$y_5 =$	0	e^3	-1	$2e^3$
$f =$	1	0	0	0

Efectuînd trei paşi Jordan începînd cu elementul pivot marcat, se ajunge la tabelul:

	$-y_4$	$-y_5$	$-y_3$	1
$y_1 =$				$1 - 2e^{-2} - 10e^{-5}$
$y_2 =$				$1 - 10e^{-3}$
$x_3 =$				10
$x_1 =$				$1 + 2e^{-2} + 10e^{-5}$
$x_2 =$				$2 + 10e^{-3}$
$f =$	$-e^{-5}$	$-e^{-5}$	$-e^{-5}$	$-1 - 2e^{-2} - 10e^{-5}$

care arată că soluţia problemei (6.9) este $\mathbf{x}^2 = (x_1^2, x_2^2, x_3^2)$, unde

$$x_1^2 = 1 + 2e^{-2} + 10e^{-5} \approx 1{,}3$$
$$x_2^2 = 2 + 10e^{-3} \approx 2{,}5$$
$$x_3^2 = 10$$

Scanned with OKEN Scanner

iar

$$f(\mathbf{x}^2) = -1 - 2e^{-2} - 10\,e^{-5} \approx -1{,}3.$$

Cu această soluţie se poate acum proceda la fel. Noi nu vom continua însă algoritmul deoarece calculele se complică. Observăm însă că valorile funcţiei de scop $f(\mathbf{x}) = -x_1$ au crescut de la pas la pas şi vor tinde [14] către soluţia exactă $-0{,}834$ care este atinsă în $x_1 = \ln\ln 10 \simeq 0{,}83403$, $x_2 = \ln 10 \simeq 2{,}30250$, $x_3 = 10$.

§ 7. Metoda paşilor mici

Cum era şi natural, s-a încercat (ARROW, K. J. HURWICZ, L. şi UZAWA, H. [4], 1958) să se dea o metodă directă, bazată pe înlocuirea în ecuaţia traiectoriilor ortogonale a derivatei cu raportul creşterilor $\Delta x/\Delta t$, considerînd, evident, Δt suficient de mic. Această metodă nu are însă o importanţă practică prea mare.

O metodă întrucîtva similară, dar mult mai importantă din punct de vedere practic a fost dată în anul 1961 de GRIFFITH, R. E. şi STEWART. R. A. [24] şi numită de aceştia *metoda M. A. P.* (iniţialele de „method of approximation programming").

Într-un anumit sens se poate spune că şi această metodă se bazează pe metoda planelor secante, cu deosebirea că aici aproximarea domeniului soluţiilor admisibile se face prin hiperplane „dinăuntru", nu din afară cum se făcea la metoda planelor secante.

Metoda poate fi descrisă în felul următor:

1°. Se porneşte de la o soluţie admisibilă arbitrară $\mathbf{x}^0$.

2°. Presupunînd că cunoaştem deja soluţiile aproximative $\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^k$, se liniarizează toate restricţiile $g_j(\mathbf{x})$ ca şi funcţia de scop $f(\mathbf{x})$. Liniarizarea aceasta făcîndu-se în raport cu punctul $\mathbf{x}^k$, se rezolvă următoarea problemă de programare liniară: *să se maximizeze funcţia*

$$\nabla f(\mathbf{x}^k)\mathbf{x}$$

în condiţiile

$$\nabla g_j(\mathbf{x}^k)\mathbf{x} \leqslant \nabla g_j(\mathbf{x}^k)\mathbf{x}^k - g_j(\mathbf{x}^k), \qquad (j = 1,2,\ldots,m),$$

$$|x_i - x_i^k| \leqslant \delta_k, \qquad (i = 1,2,\ldots,n),$$

unde δ_k este un număr pozitiv suficient de mic, care determină lungimea pasului, numită problema LP_k.

3°. Se repetă paşii de la 2°, pînă cînd funcţia de scop variază de la pas la pas cu valori neglijabile.

Scanned with OKEN Scanner

Principalul avantaj al acestei metode constă în faptul că *se poate aplica și în cazul unei programări neconvexe.* De asemenea, această metodă se poate programa relativ ușor la mașină și fiind liniarizată i se poate aplica metoda simplex.

Ca dezavantaje menționăm faptul că nu se dă nici o metodă pentru determinarea lungimii pasului (a mărimii numărului δ_k). De asemenea nu este demonstrată convergența metodei și nici existența punctului de optim.

Am prezentat totuși această metodă deoarece este prima încercare de a se da o metodă valabilă și în cazul unei probleme de programare neliniare oarecare.

Exemplu:

$$\max f(\mathbf{x}) = \max (2x_1 + x_2)$$

în condițiile

$$g_1(\mathbf{x}) = x_1^2 + x_2^2 - 25 \leqslant 0$$
$$g_2(\mathbf{x}) = x_1^2 - x_2^2 - 7 \leqslant 0$$
$$x_1 \geqslant 0,\ x_2 \geqslant 0.$$

Ca punct de pornire se ia soluția admisibilă $\mathbf{x}^0 = (1,1)$. Avem

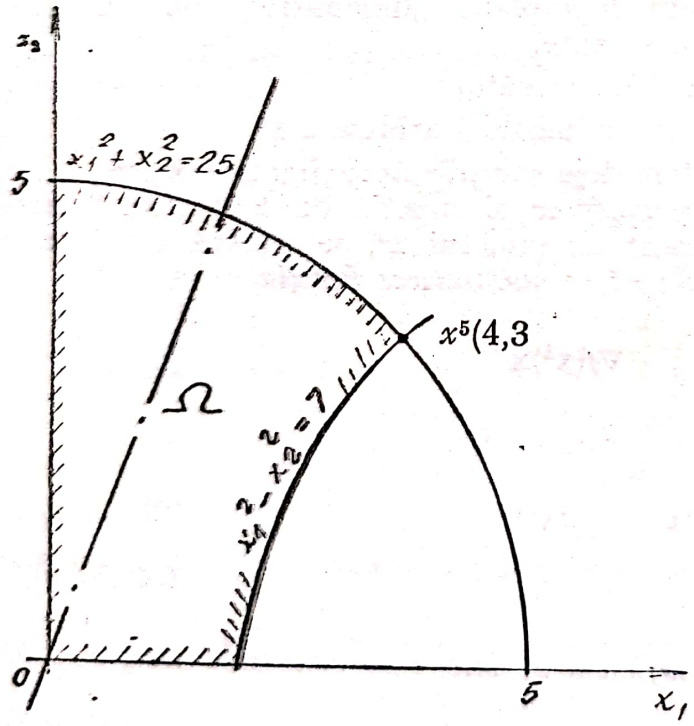

Figura 7.1

$$\nabla f(\mathbf{x}^0) = (2,\ 1),$$
$$\nabla g_1(\mathbf{x}^0) = (2,\ 2)$$
$$\nabla g_2(\mathbf{x}^0) = (2,\ -2),$$
$$g_1(\mathbf{x}^0) = -23,\ g_2(\mathbf{x}^0) = -7.$$

Prin urmare, se rezolvă următoarea problemă de programare liniară LP_1:

$$f(\mathbf{x}) = 2x_1 + x_2 \to \max$$

în condițiile:

$$2(x_1 - 1) + 2(x_2 - 1) \leqslant 23$$
$$2(x_1 - 1) - 2(x_2 - 1) \leqslant 7$$
$$-1 \leqslant x_1 - 1 \leqslant 1$$
$$-1 \leqslant x_2 - 1 \leqslant 1$$
$$x_1 \geqslant 0,\ x_2 \geqslant 0.$$

Scanned with OKEN Scanner

Tabelul simplex corespunzător este

	$-x_1$	$-x_2$	1
$y_1 =$	2	2	27
$y_2 =$	2	-1	7
$y_3 =$	1	0	2
$y_4 =$	0	$\boxed{1}$	2
$f =$	-2	-1	0

Efectuînd doi paşi Jordan începînd cu elementul pivot marcat, se ajunge la tabelul

	$-y_3$	$-y_4$	1
$y_1 =$			19
$y_2 =$			5
$x_1 =$			2
$x_2 =$			2
$f =$	2	1	6

care arată că $\mathbf{x}^1 = (2,2)$ este soluţia problemei LP_1 şi $f(\mathbf{x}^1) = 6$.

În punctul $\mathbf{x}^1$ avem

$$\nabla g_1(\mathbf{x}^1) = (4,4)$$

$$\nabla g_2(\mathbf{x}^1) = (4, -4)$$

$$g_1(\mathbf{x}^1) = 17,\ g_2(\mathbf{x}^1) = -7.$$

Următoarea problemă de programare liniară LP_2 este:

$$f(\mathbf{x}) = 2x_1 + x_2 \to \max$$

în condiţiile

$$4(x_1 - 2) + 4(x_2 - 2) \leqslant 17$$

$$4(x_1 - 2) - 4(x_2 - 2) \leqslant 7$$

$$-1 \leqslant x_1 - 2 \leqslant 1$$

$$-1 \leqslant x_2 - 2 \leqslant 1, \qquad x_1 \geqslant 0, \qquad x_2 \geqslant 0.$$

Scanned with OKEN Scanner

Tabelul simplex corespunzător problemei LP_2 este:

	$-x_1$	$-x_2$	1
$y_1 =$	4	4	33
$y_2 =$	4	−4	7
$y_3 =$	1	0	3
$y_4 =$	$\boxed{-1}$	0	−1
$y_5 =$	0	1	3
$y_6 =$	0	−1	−1
$f =$	−2	−1	0

După efectuarea a doi pași Jordan modificați începînd cu elementul pivot marcat se ajunge la tabelul:

	$-y_4$	$-y_6$	1
$y_1 =$	4	4	25
$y_2 =$	4	−4	7
$y_3 =$	1	0	2
$x_1 =$	−1	0	1
$y_5 =$	0	$\boxed{1}$	2
$x_2 =$	0	−1	1
$f =$	−2	−1	3

de pe care se poate citi o soluție admisibilă. După efectuarea a încă doi pași Jordan modificați începînd cu elementul pivot marcat, se va ajunge la tabelul:

	$-y_3$	$-y_5$	1
$y_1 =$			9
$y_2 =$			7
$y_4 =$			2
$x_1 =$			3
$y_6 =$			2
$x_2 =$			3
$f =$	2	1	9

care arată că soluția problemei LP_2 este $\mathbf{x}^2 = (3,3)$ și $f(\mathbf{x}^3) = 9$.

Scanned with OKEN Scanner

În acest punct avem

$$\nabla g_1(\mathbf{x}^2) = (6,6), \qquad \nabla g_2(\mathbf{x}^2) = (6, -6)$$
$$g_1(\mathbf{x}^2) = -7, \qquad g_2(\mathbf{x}^2) = -7.$$

Deci următoarea problemă de programare liniară LP_3 este

$$\max f(\mathbf{x}) = \max (2x_1 + x_2)$$

în condițiile

$$6(x_1 - 3) + 6(x_2 - 3) \leqslant 7$$
$$6(x_1 - 3) - 6(x_2 - 3) \leqslant 7$$
$$-1 \leqslant x_1 - 3 \leqslant 1$$
$$-1 \leqslant x_2 - 3 \leqslant 1, \qquad x_1 \geqslant 0, \qquad x_2 \geqslant 0.$$

Soluția problemei LP_3 este $\mathbf{x}^3 = \left(4, \frac{17}{6}\right)$ și

$$f(\mathbf{x}^3) = \frac{65}{6} = 10{,}833.$$

Următoarea problemă LP_4 este:

$$\max f(\mathbf{x}) = \max (2x_1 + x_3)$$

supusă condițiilor

$$8(x_1 - 4) + \frac{17}{3}\left(x_2 - \frac{17}{6}\right) \leqslant \frac{35}{36}$$
$$8(x_1 - 4) - \frac{17}{3}\left(x_2 - \frac{17}{6}\right) \leqslant -\frac{35}{36}$$
$$-1 \leqslant x_1 - 4 \leqslant 1$$
$$-1 \leqslant x_2 - \frac{17}{6} \leqslant 1, \qquad x_1 \geqslant 0, \qquad x_2 \geqslant 0$$

Soluția acestei probleme este $\mathbf{x}^4 = (3{,}993\,;\ 3{,}025)$ și $f(\mathbf{x}^4) = 11{,}011$.

După încă un pas, rezolvînd problema LP_5 se găsește $\mathbf{x}^5 = (4,3)$ și și $f(\mathbf{x}^5) = 11$, care este, după cum se vede din Figura 7.1, soluția exactă a problemei inițiale.

Observație 7.1. Soluția aproximativă $\mathbf{x}^4$ nu este o soluție admisibilă a problemei inițiale, deoarece a doua restricție nu este verificată.

Scanned with OKEN Scanner

§ 8. Metoda pașilor mari

Una din metodele cele mai des utilizate este metoda cunoscută sub denumirea de *metoda pașilor mari*. Există o serie de variante ale acestei metode. Primele variante ale acestei metode au fost date de FRISCH, R.A.K., [20], ROSEN, J. B. [67] și ZOUTENDIJK, G. [82]. Această metodă se referă la următoarea problemă de programare neliniară: *să se maximizeze funcția* f *în condițiile* $\mathbf{Ax} \leqslant \mathbf{b}$, *unde* f *este o funcție concavă și admite derivate parțiale de ordinul întîi continui în domeniul soluțiilor admisibile.*

Și în această metodă se pornește de la un punct arbitrar $\mathbf{x}^0$ din domeniul soluțiilor, apoi se trece de la iterația a k—a la iterația $(k+1)$— a, determinînd o direcție $s^k=(s_1, s_2, \ldots, s_n)'$ și un număr pozitiv t (lungimea pasului), astfel încît:

$$\mathbf{x}^k + t\mathbf{s}^k \in \Omega, \tag{8.1}$$

unde Ω este domeniul soluțiilor admisibile ale problemei puse, adică

$$\Omega = \{\mathbf{x} \in \mathbf{R}^n | \ \mathbf{Ax} \leqslant \mathbf{b}\}.$$

Condiția (8.1) se scrie și sub forma

$$\mathbf{a}_j\mathbf{s}^k \leqslant 0, \tag{8.2}$$

pentru toți j pentru care $\mathbf{a}_j\mathbf{x}^k = b_j$, unde

$$\mathbf{a}_j = (a_{j1}, a_{j2}, \ldots, a_{jn}).$$

O direcție $\mathbf{s}^k$ care verifică condițiile (8.2) se numește *direcție admisibilă*.

Pentru ca funcția de scop f să crească pe această direcție este necesar să avem

$$\nabla f(\mathbf{x}^k)\,\mathbf{s}^k > 0.$$

Dacă direcția $\mathbf{s}^k$ a fost deja determinată, atunci se notează cu t' cea mai mică distanță de la punctul $\mathbf{x}^k$ la frontiera domeniului soluțiilor, pe direcția $\mathbf{s}$, iar cu t'' valoarea corespunzătoare lui t pentru care funcția de scop $f(\mathbf{x}^k + t\mathbf{s}^k)$ își atinge maximul.

Fie

$$t^k = \min(t', t'').$$

Atunci următorul punct va fi

$$\mathbf{x}^{k+1} = \mathbf{x}^k + t^k\mathbf{s}^k.$$

Scanned with OKEN Scanner

Se obţin diverse variante ale acestei metode dacă se precizează modul de alegere a direcţiei admisibile $\mathbf{s}^k$. De exemplu FRISCH, în lucrarea amintită, ia pentru direcţia admisibilă $\mathbf{s}^k$ o perpendiculară la frontiera domeniului (hiperplan sau muchie) de partea în care funcţia de scop creşte. ROSEN, J. B. [67] ia pentru direcţia admisă $\mathbf{s}^k$ proiecţia gradientului funcţiei de scop în $\mathbf{x}^k$ pe varietatea formată din toate hiperplanele de restricţie care trec prin punctul $\mathbf{x}^k$.

O altă variantă a acestei metode a fost dată de ZOUTENDIJK în lucrarea amintită mai sus, variantă care s-a dovedit a fi foarte eficace, alegînd în punctul $\mathbf{x}^k$ direcţia optimă de creştere a funcţiei de scop. Găsirea acestei direcţii optime se face prin rezolvarea unei *probleme „mici"* de programare liniară, care se ataşează punctului $\mathbf{x}^k$. Astfel se rezolvă următoarea problemă de programare liniară: *să se maximizeze funcţia*

$$\nabla f(\mathbf{x}^k)\mathbf{s}$$

în condiţiile

$$\mathbf{a}_j\mathbf{s} \leqslant 0, \qquad \forall j \in J,$$

unde J reprezintă mulţimea indicilor j pentru care $g_j(\mathbf{x}^k) = 0$.

Autorul numeşte această problemă *„problemă mică"*, întrucît în general numărul restricţiilor este mic, reprezentînd numărul hiperplanelor ce trec prin punctul $\mathbf{x}^k$. În realitate, acestor restricţii trebuie să li se mai adauge şi o condiţie de mărginire pentru vectorul $\mathbf{s}$, deoarece fiind vorba de maxim domeniul restricţiilor trebuie să fie mărginit. Ori această mărginire mai adaugă încă n condiţii în plus, aşa că numărul total de restricţii va creşte cu inegalităţile de mărginire a componentelor lui $\mathbf{s}$.

Alte două variante ale acestei metode au fost date de MARUŞCIAC I. şi RĂDULESCU M. [49] – [52], în care se indică un procedeu relativ comod de găsire a unei direcţii admisibile, folosind metoda proiecţiei gradientului, fie pe un hiperplan, fie pe varietatea comună care conţine punctul $\mathbf{x}^k$. Vom descrie aici una din aceste două variante.

Se consideră următoarea problemă de programare convexă: *să se minimizeze funcţia convexă*

$$f(\mathbf{x})$$

în condiţiile

$$\mathbf{A}\mathbf{x} \leqslant \mathbf{b}.$$

Şi în metoda noastră se porneşte de la un punct arbitrar $\mathbf{x}^0$ al domeniului soluţiilor Ω şi se construiesc aproximaţiile succesive tot cu metoda pasului mare. Alegerea direcţiei însă se face în felul următor: Din fiecare punct $\mathbf{x}^k$ se merge fie pe direcţia gradientului, fie pe proiecţia sa pe unul din

Scanned with OKEN Scanner

hiperplanele ce trec prin acel punct. Numai în cazul cînd nici una din aceste proiecţii nu dă o direcţie admisibilă se recurge la proiecţia gradientului pe muchia comună ce conţine punctul $\mathbf{x}^k$. În felul acesta se simplifică la maximum iteraţiile, deoarece proiecţia gradientului pe un hiperplan se scrie foarte uşor. Sînt însă cazuri cînd aceste proiecţii nu conduc la direcţii admisibile şi atunci trebuie recurs la proiecţia pe varietatea comună a acestor hiperplane. Aceasta din urmă este totdeauna admisibilă, cu excepţia cazului cînd punctul $\mathbf{x}^k$ este punctul optimal căutat. De obicei acest pas intervine în apropierea punctului optim, către sfîrşitul algoritmului.

Metoda poate fi descrisă în felul următor:

1°. Se porneşte de la un punct arbitrar $\mathbf{x}^0$ din domeniul soluţiilor Ω.

2°. Fie $\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^k$ aproximaţiile succesive deja construite. Dacă în punctul $\mathbf{x}^k$ avem

$$\mathbf{a}_j\mathbf{x}^k = b_j, \qquad j = 1,2, \ldots, p; \qquad \mathbf{a}_j\mathbf{x}^k \leqslant b_j, \qquad i > p. \tag{8.3}$$

Se proiectează gradientul $\nabla f(\mathbf{x}^k)$ mai întîi pe hiperplanul

$$\mathbf{a}_1\mathbf{x} = b_1.$$

Dacă această direcţie este admisă, atunci se construieşte iteraţia următoare după metoda descrisă mai sus. În caz contrar se trece la proiecţia pe hiperplanul

$$\mathbf{a}_2\mathbf{x} = b_2$$

cu care se procedează la fel.

Dacă nici una din aceste proiecţii nu furnizează o direcţie admisă, atunci se trece la proiecţia gradientului $\nabla f(\mathbf{x}^k)$ pe muchia comună planelor

$$\mathbf{a}_j\mathbf{x} = b_j, \qquad (j = 1,2, \ldots, p). \tag{8.4}$$

Se arată că această proiecţie sau este o direcţie admisibilă, sau punctul $\mathbf{x}^k$ este un punct optimal. În cazul cînd $p > n$, trebuie efectuată proiecţia pe fiecare muchie determinată de cîte $n - 1$ hiperplane din cele p hiperplane (8.4) care trec prin punctul $\mathbf{x}^k$.

În lucrarea [49] se demonstrează că acest procedeu este convergent, adică

$$\lim_{k\to\infty} \mathbf{x}^k = \mathbf{x}^*,$$

unde $\mathbf{x}^*$ este soluţia optimă căutată.

Se arată de asemenea cum se poate evita cazul cînd algoritmul prezintă un număr infinit de paşi. Astfel se arată că algoritmul este infinit numai

Scanned with OKEN Scanner

şi numai atunci cînd pentru $k > k_0$ toate iteraţiile se găsesc într-o aceeaşi varietate liniară (muchie). De aceea, practic după ce se constată acest lucru, se poate întrerupe algoritmul, rezolvîndu-se apoi problema minimului funcţiei $f(\mathbf{x})$ cu legături. Legăturile fiind tocmai ecuaţiile hiperplanelor ce determină varietatea respectivă.

Pentru determinarea proiecţiilor se dau formule explicite. Astfel proiecţia $\mathbf{u}^i$ a vectorului $\mathbf{v}$ pe hiperplanul

$$\mathbf{a}_i\mathbf{x} = b_i$$

este dată de relaţiile

$$\mathbf{u}^i = \mathbf{v} + \mu_i \mathbf{a}_i ,$$

unde

$$\mu_i = -\frac{\mathbf{a}_i \mathbf{v}}{\mathbf{a}_i \mathbf{a}_i'}$$

Formule asemănătoare se dau şi pentru proiecţia pe o muchie comună a mai multor hiperplane a unui vector, în care intervin determinanţi ai lui Gramm în raport cu vectorii $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_p$.

În cadrul metodei paşilor mari pentru cazul general, un loc important îl ocupă o metodă dată în două variante de ZUHOVIŢKI, S. I. şi AVDEEVA, L. I [86]. Avînd în vedere importanţa ei practică, o vom prezenta în cele ce urmează ceva mai în detaliu.

Astfel, se consideră următoarea problemă de programare convexă: *să se minimizeze funcţia*

$$f(\mathbf{x}) = f(x_1, x_2, \ldots x_n) \tag{8.5}$$

în condiţiile

$$g_j(\mathbf{x}) = g_j(x_1, x_2, \ldots, x_n) \leqslant 0, \qquad (j = 1, 2, \ldots, m), \tag{8.6}$$

unde f şi g_j, $j = 1, 2, \ldots, m$, sînt funcţii convexe netede (adică admit derivate parţiale continue), *iar domeniul soluţiilor Ω are puncte interioare, adică puncte în care*

$$g_j(\mathbf{x}) < 0, \qquad (j = 1, 2, \ldots, m).$$

După cum am văzut, această problemă se poate reduce totdeauna la o formă canonică, în care funcţia de scop este liniară, introducînd încă o variabilă independentă x_{n+1} şi restricţia

$$f(\mathbf{x}) \leqslant x_{n+1}.$$

Scanned with OKEN Scanner

Atunci se poate considera următoarea problemă de programare convexă canonică: *să se minimizeze funcţia*

$$f(\mathbf{x}) = \mathbf{cx} \tag{8.7}$$

în condiţiile

$$g_j(\mathbf{x}) \leqslant 0. \qquad (j = 1, 2, \ldots, m). \tag{8.8}$$

Metoda propusă de Zuhoviţki constă în determinarea unor direcţii admisibile optime în raport cu un şir (δ_k) *de „coeficienţi de monotonie"* pentru funcţia de scop, concretizat într-un anumit şir descrescător de numere pozitive.

1. Varianta întîi a algoritmului lui Zoutendijk — Zuhoviţki

Algoritmul constă din două părţi. În prima parte se găseşte un punct din Ω, adică o soluţie admisibilă, iar în a doua, îmbunătăţirea soluţiei şi găsirea la limită a soluţiei optime.

Pentru găsirea unei soluţii admisibile a problemei (8.7) — (8.8) se poate folosi chiar algoritmul descris în partea a doua, aplicîndu-l problemei: *să se minimizeze funcţia*

$$u = \xi$$

în condiţiile

$$g_j(\mathbf{x}) \leqslant \xi, \qquad (j = 1, 2, \ldots, m)$$

care determină un domeniu Ω'.

Întrucît prin ipoteză domeniul Ω conţine puncte interioare rezultă că

$$\min_{(\mathbf{x}, \xi) \in \Omega'} u < 0.$$

De aceea, pentru a obţine o soluţie admisibilă a problemei (8.7)—(8.8) nu este necesar să se găsească minimul funcţiei u, ci e suficient să se oprească procesul algoritmic în momentul cînd s-a determinat un număr ξ_0, pentru care $u = \xi_0 < 0$, deoarece atunci punctul $\mathbf{x}^0$ corespunzător aparţine lui Ω.

Partea a doua a algoritmului constă în repetarea unui pas general adică în:

1) determinarea direcţiei admisibile optime pentru punctul $\mathbf{x}^k$ şi funcţia de scop f;

Scanned with OKEN Scanner

2) calculul noului număr δ_k (parametrul monotoniei);
3) calculul pasului $t_k > 0$ corespunzător lungimii segmentului admisibil după direcţia optimă găsită;
4) determinarea noului punct (de aproximaţie succesivă) $\mathbf{x}^{k+1}$.

Pasul general. Să presupunem că se cunoaşte deja o soluţie admisibilă $\mathbf{x}^0 \in \Omega$. Fără a restrînge generalitatea putem presupune că $\mathbf{x}^0$ se găseşte pe frontiera lui Ω, căci în caz contrar putem ajunge la un asemenea punct deplasîndu-ne din $\mathbf{x}^0$ de-a lungul vectorului $-\mathbf{c} = -(c_1, c_2, \ldots, c_n)$, care revine apoi la găsirea celei mai mici rădăcini pozitive dintre rădăcinile ecuaţiilor

$$g_j(\mathbf{x}^0 - t\mathbf{c}) = 0, \qquad (j = 1, 2, \ldots, m).$$

Pentru a evita aşa numitul fenomen de gripare (înţepenire) datorat erorilor care se acumulează, pus în evidenţă şi de ZOUTENDIJK, G. [82], şi pentru delimitarea erorilor de calcul care pot surveni în procesul calculelor, Zuhoviţki introduce la fiecare pas un parametru pozitiv δ_k, începînd cu un număr arbitrar suficient de mic, $\delta_1 > 0$, la primul pas.

Astfel, un punct $\mathbf{x}^0$ se consideră pe suprafaţa $g_{j_0}(\mathbf{x}) = 0$, dacă

$$-\delta_1 < g_{j_0}(\mathbf{x}^0) \leqslant 0,$$

adică abaterea punctului $\mathbf{x}^0$ de la suprafaţa $g_{j_0}(\mathbf{x}) = 0$ este nepozitivă şi mai mare de $-\delta_1$.

Să presupunem, de exemplu, că în punctul $\mathbf{x}^0$ avem

$$-\delta_1 < g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots, p)$$

$$g_j(\mathbf{x}^0) < \delta_1, \qquad (j = p+1, \ldots, m).$$

1) *Determinarea direcţiei admisibile optime în punctul* $\mathbf{x}^0$. Direcţia $\mathbf{s}^0$ se va determina astfel încît pe această direcţie funcţia de scop

$$f(\mathbf{x}) = \mathbf{cx}$$

să descrească, adică

$$\mathbf{cs}^0 < \mathbf{0}, \tag{8.9}$$

şi să pătrundă în interiorul domeniului Ω (în sens strict), adică pe această direcţie să descrească toate funcţiile g_j, $j = 1, 2, \ldots, p$. Deci să aibă loc inegalităţile

$$\nabla g_j(\mathbf{x}^0)\mathbf{s}^0 < \mathbf{0}, \qquad (j = 1, 2, \ldots, p). \tag{8.10}$$

Scanned with OKEN Scanner

Pentru a determina direcţia celei mai rapide scufundări este necesar ca cea mai mică în valoare absolută dintre mărimile

$$\mathbf{cs}^0, \qquad \nabla g_j(\mathbf{x}^0)\mathbf{s}^0, \qquad (j = 1, 2, \ldots, p) \tag{8.11}$$

să fie cît mai mare. Deoarece toate aceste mărimi sînt negative aceasta înseamnă că cea mai mare dintre mărimile (8.11) să fie minimă. Cu alte cuvinte, direcţia $\mathbf{s}^0$ trebuie să fie cebîşeviană, adică $\mathbf{s}^0$ este punctul lui Cebîşev al sistemului de inegalităţi

$$\mathbf{cs} < 0$$

$$g_j(\mathbf{x}^0)\mathbf{s} < 0, \qquad (j = 1, 2, \ldots, p).$$

Pentru găsirea punctului cebîşevian se rezolvă următoarea problemă de minim: *să se minimizeze funcţia*

$$u = \eta \tag{8.12}$$

în condiţiile

$$\mathbf{cs} \leqslant \eta, \quad \nabla g_j(\mathbf{x}^0)\mathbf{s} \leqslant \eta, \qquad (j = 1,2, \ldots, m)$$

$$|s_1| \leqslant C, \quad |s_2| \leqslant C, \ldots, |s_n| \leqslant C, \tag{8.13}$$

unde $\mathbf{s}' = (s_1, s_2, \ldots, s_n)$ şi C este un număr oarecare, suficient de mare. Ultimele inegalităţi din (8.13) au fost necesare pentru ca domeniul soluţiilor sistemului de inegalităţi determinat de primele două să fie mărginit.

Să notăm cu η_1 minimul funcţiei u în condiţiile (8.13).

2) *Calculul noului parametru* δ. Se deosebesc două cazuri distincte:

a) $\eta_1 < -\delta_1$ şi b) $\eta_1 \geqslant -\delta_1$.

În cazul a) nu se schimbă parametrul, adică se ia

$$\delta_2 = \delta_1.$$

În cazul b) se modifică valoarea parametrului δ după cum urmează: dacă $-\delta_1 \leqslant \eta_1 < 0$, atunci se pune

$$\delta_2 = \frac{\delta_1}{2}$$

şi procesul se continuă, deplasîndu-ne de-a lungul direcţiei $\mathbf{s}^0$, pînă se ajunge la un nou punct de aproximaţie.

Scanned with OKEN Scanner

Dacă

$$\eta_1 = 0 \quad \text{şi}$$

$$g_{j_k}(\mathbf{x}^0) = 0 \quad (k = 1, 2, \ldots, q), \quad g_j(\mathbf{x}^0) < 0 \quad (j \neq j_1, \ldots, j_q),$$

atunci se rezolvă următoarea problemă de programare liniară: *să se minimizeze funcţia*

$$u = \eta \tag{8.14}$$

în condiţiile

$$\mathbf{es} \leqslant \eta, \quad g_{j_k}(\mathbf{x}^0)\mathbf{s} \leqslant \eta, \qquad (k = 1, 2, \ldots, q) \tag{8.15}$$

$$|s_i| \leqslant C, \qquad (i = 1, 2, \ldots, n)$$

Dacă

$$\min u = \eta_1' = 0,$$

atunci $\mathbf{x}^0$ este soluţia problemei iniţiale, şi procesul algoritmic se termină.

Dacă însă $\min u = \eta_1' < 0$, atunci se pune

$$\delta_2 = \frac{\delta_1}{2}$$

şi se continuă procesul, deplasîndu-ne de-a lungul direcţiei $\mathbf{s}^0$, soluţia problemei (8.14)—(8.15).

3) *Calculul pasului t.* Să presupunem că se cunoaşte deja direcţia optimă $\mathbf{s}^0$ şi valoarea parametrului δ_2. Atunci se găseşte cea mai mică rădăcină pozitivă dintre rădăcinile ecuaţiilor:

$$g_j(\mathbf{x}^0 + t\,\mathbf{s}^0) = \mathbf{0}, \qquad (j = 1, 2, \ldots, m).$$

Fie t_1 această rădăcină.

4) *Determinarea noului punct de aproximaţie.* Noul punct de aproximaţie va fi punctul

$$\mathbf{x}^1 = \mathbf{x}^0 + t_1\mathbf{s}^0.$$

Cu punctul $\mathbf{x}^1$ se procedează apoi la fel ca şi cu punctul $\mathbf{x}^0$ luînd însă ca parametru δ_2.

Scanned with OKEN Scanner

Criteriul de optim. Să presupunem că s-a determinat aproximaţia succesivă $\mathbf{x}^{k-1}$ şi în determinarea direcţiei $\mathbf{s}^{k-1}$ s-a constatat că

$$\min u = \eta_k = 0,$$

şi de asemenea problema de programare liniară corespunzătoare lui $\mathbf{x}^{k-1}$ de tipul (8.14)—(8.15) are ca soluţie

$$\min u = \eta_k' = 0.$$

Vom arăta că atunci *punctul* $\mathbf{x}^{k-1}$ *este soluţie optimă.* Într-adevăr, din faptul că $\eta_k = 0$, rezultă că nu există o direcţie admisă din punctul $\mathbf{x}^{k-1}$. Aceasta însă nu înseamnă că $\mathbf{x}^{k-1}$ este un punct de optim, deoarece s-ar putea ca neexistenţa unei direcţii admisibile să se datoreze faptului că parametrul δ_k nu este suficient de mic, din care cauză pot exista suprafeţe pe care nu este situat punctul $\mathbf{x}^{k-1}$ însă care sînt mai „aproape" decît δ_k de punctul $\mathbf{x}^{k-1}$.

Dacă însă în plus are loc şi egalitatea

$$\eta_k' = 0,$$

atunci inexistenţa direcţiei admisibile (de scufundare) este consecinţa faptului că punctul $\mathbf{x}^{k-1}$ este un punct de optim, deoarece în problema de programare liniară (8.14)—(8.15) participă numai acele suprafeţe de restricţie pe care se află efectiv punctul $\mathbf{x}^{k-1}$.

ZOUTENDIJK G. [81] a dat o metodă prin care se poate da o evaluare a abaterii unui punct aproximativ $\mathbf{x}^k$ faţă de punctul optimal $\mathbf{x}^*$.

Avînd în vedere importanţa practică a unei asemenea evaluări vom arăta în cele ce urmează la ce problemă de programare liniară conduce.

Astfel, ecuaţia planului tangent la suprafaţa $z = g_j(\mathbf{x})$ în punctul $\mathbf{x} = \mathbf{x}^k$ are forma

$$Z - g_j(\mathbf{x}^k) = \nabla g_j(\mathbf{x}^k)\,(\mathbf{x} - \mathbf{x}^k).$$

Deoarece funcţiile g_j sînt convexe, avem $Z \leqslant g_j(\mathbf{x})$ şi obţinem inegalitatea

$$\nabla g_j(\mathbf{x}^k)\,(\mathbf{x} - \mathbf{x}^k) \leqslant g_j(\mathbf{x}) - g_j(\mathbf{x}^k),$$

care, întrucît pentru orice $\mathbf{x} \in \Omega$, avem $g_j(\mathbf{x}) \leqslant 0$, conduce la inegalitatea

$$\nabla g_j(\mathbf{x}^k)\,(\mathbf{x} - \mathbf{x}^k) + g_j(\mathbf{x}^k) \leqslant 0, \qquad (j = 1, 2, \ldots, m).$$

Scanned with OKEN Scanner

Prin aceasta s-a stabilit că domeniul Ω este conţinut în întregime în domeniul Ω_1, definit de ultimele inegalităţi liniare.

Astfel, deoarece $\Omega \subset \Omega_1$ avem:

$$\min_{\mathbf{x}\in\Omega_1} (f(\mathbf{x}) - f(\mathbf{x}^k)) \leqslant \min_{\mathbf{x}\in\Omega} (f(\mathbf{x}) - f(\mathbf{x}^k)) \leqslant 0.$$

Prin urmare, avem:

$$|\min_{\mathbf{x}\in\Omega} f(\mathbf{x}) - f(\mathbf{x}^k))| \leqslant |\min_{\mathbf{x}\in\Omega_1} (f(\mathbf{x}) - f(\mathbf{x}^k))|.$$

Astfel, pentru evaluarea superioară a abaterii valorii funcţiei de scop în punctul $\mathbf{x}^k$ de valoarea acesteia în punctul de optim, este suficient să rezolvăm următoarea problemă de programare liniară: *să se minimizeze funcţia*

$$f_1(\mathbf{x}) = f(\mathbf{x}) - f(\mathbf{x}^k)$$

în condiţiile

$$\nabla g_j(\mathbf{x}^k)(\mathbf{x} - \mathbf{x}^k) + g_j(\mathbf{x}^k) \leqslant 0, \qquad (j = 1, 2, \ldots, m).$$

Convergenţa algoritmului. Vom demonstra convergenţa algoritmului în ipoteza că domeniul Ω este mărginit şi închis. Atunci şirul $(f(\mathbf{x}^k))$ este un şir monoton şi mărginit şi prin urmare convergent către un număr real α.

Vom arăta mai întîi că din:

$$\lim_{k\to\infty} f(\mathbf{x}^k) = \alpha > -\infty$$

rezultă că şirul monoton descrescător de numere pozitive (δ_k) tinde către zero:

$$\lim_{k\to\infty} \delta_k = \delta = 0.$$

Într-adevăr, să presupunem contrariul. Atunci ar exista un rang k_0, astfel încît pentru orice $k > k_0$, rezultă

$$\delta_k = \delta \text{ şi } \eta_k < -\delta$$

Scanned with OKEN Scanner

Pe de altă parte, şirul aproximaţiilor succesive $(\mathbf{x}^k)$ fiind un şir mărginit admite cel puţin un punct de acumulare $\tilde{\mathbf{x}}$. Fie $\mathbf{x}^{k_i} \to \tilde{\mathbf{x}}$. Evident că $\tilde{\mathbf{x}}$ este un punct frontieră. Să presupunem că în acest punct avem

$$-\delta < g_{j_r}(\tilde{\mathbf{x}}) \leqslant 0, \qquad (r = 1, 2, \ldots, r_1)$$

(8.16)

$$g_j(\tilde{\mathbf{x}}) \leqslant -\delta, \qquad (j \neq j_1, \ldots, j_{r_1}).$$

Din cauza continuităţii funcţiilor g_j există o vecinătate $V_{\tilde{\mathbf{x}}}$ a punctului $\tilde{\mathbf{x}}$, astfel încît pentru orice $\mathbf{x} \in V_{\tilde{\mathbf{x}}} \cap \Omega$ vor fi îndeplinite inegalităţile

(8.17) $$-\delta < g_{j_r}(\mathbf{x}) \leqslant 0, \qquad (r = 1, 2, \ldots, r_1).$$

Deoarece $\mathbf{x}^{k_i} \to \tilde{\mathbf{x}}$, va exista un număr $k_1 \geqslant k_0$ astfel încît pentru $k_i > k_1$, rezultă $\mathbf{x}^{k_i} \in V_{\tilde{\mathbf{x}}} \cap \Omega$, aşa încît la determinarea direcţiei admisibile din punctul $\mathbf{x}^{k_i}$ vor participa toate funcţiile $g_{j_1}, g_{j_2}, \ldots, g_{j_{r_1}}$.

În continuare, pentru orice $\mathbf{x}^{k_i} \in V_{\tilde{\mathbf{x}}} \cap \Omega$ au loc inegalităţile

$$\mathbf{e}\mathbf{s}^{k_i} \leqslant \eta_{k_i} < -\delta,$$

(8.18)

$$\nabla g_{j_r}(\mathbf{x}^{k_i})\mathbf{s}^{k_i} \leqslant \eta_{k_i} < -\delta, \qquad (r = 1, 2, \ldots, r_1).$$

Deoarece şirul (s^{k_i}) este mărginit, din el se poate extrage un subşir convergent. Pentru simplificarea notaţiei se poate presupune că

$$\lim_{i \to \infty} \mathbf{s}^{k_i} = \tilde{\mathbf{s}}.$$

Trecînd la limită în inegalităţile (8.18) pentru $i \to \infty$, obţinem

$$\mathbf{e}\tilde{\mathbf{s}} < -\delta, \quad \nabla g_{j_r}(\tilde{\mathbf{x}})\mathbf{s} < -\delta, \qquad (r = 1, 2, \ldots, r_1).$$

Prin urmare, direcţia $\tilde{\mathbf{s}}$ este o direcţie admisă din punctul $\tilde{\mathbf{x}}$. Mărimea pasului după această direcţie este un număr pozitiv $\tilde{t} > 0$. De aceea, avem:

$$g_j\left(\tilde{\mathbf{x}} + \frac{\tilde{t}}{2}\tilde{\mathbf{s}}\right) < \mathbf{0}, \qquad (j = 1, 2, \ldots, m)$$

Scanned with OKEN Scanner

care arată că pentru k_i suficient de mare vom avea de asemenea

$$g_j\left(\mathbf{x}^{k_i} + \frac{\tilde{t}}{2}\mathbf{s}^{k_i}\right) < \mathbf{0}, \qquad (j = 1, 2, \ldots, m).$$

Prin urmare, pentru k_i suficient de mare vom avea:

$$t_{k_i} > \frac{\tilde{t}}{2}.$$

Ținînd seamă de aceste observații, avem în consecință relațiile:

$$f(\mathbf{x}^{k_i}) = f\left(\mathbf{x}^{k_i-1} + t_{k_i}\mathbf{s}^{k_i-1}\right) = f\left(\mathbf{x}^{k_i-1}\right) + f\left(t_{k_i}\mathbf{s}^{k_i-1}\right) =$$

$$= f\left(\mathbf{x}^{k_i-2}\right) + f\left(t_{k_i-1}\mathbf{s}^{k_i-2}\right) + f\left(t_{k_i}\mathbf{s}^{k_i-1}\right) =$$

. . . . . .

$$= f(\mathbf{x}^0) + \sum_{j=1}^{k_i} f(t_j\mathbf{s}^{j-1}) = f(\mathbf{x}^0) + \sum_{j=1}^{k_i} t_j \sum_{h=1}^{n} c_h s_h^{j-1} \leqslant$$

$$\leqslant f(\mathbf{x}^0) + \sum_{j}^{i} t_{k_j}\mathbf{c}\,\mathbf{s}^{k_j-1} \leqslant f(\mathbf{x}^0) - \frac{\tilde{t}}{2} i\delta,$$

adică

$$\lim_{i\to\infty} f(\mathbf{x}^{k_i}) = -\infty.$$

Aceasta contrazice însă ipoteza că $\lim_{i\to\infty} f(\mathbf{x}^{k_i}) > -\infty$.

Prin urmare

$$\lim_{k\to\infty} \delta_k = \delta = 0.$$

Vom arăta acum că șirul $(f(\mathbf{x}^k))$ este un șir convergent către α. Pentru aceasta, vom nota cu k_i' numărul pasului la care trebuie schimbat parametrul δ_k. Atunci din regula de construire a parametrilor δ_k, deducem că

$$\lim_{k_i'\to\infty} \eta_{k_i'} = 0.$$

Scanned with OKEN Scanner

Fie $(\mathbf{x}^{k'_i})$ şirul corespunzător paşilor k'_i. Deoarece şi acesta este un şir mărginit, rezultă că din el se poate extrage un şir convergent către $\mathbf{x}^*$. Putem presupune că avem

$$\lim_{k'_i \to \infty} \mathbf{x}^{k'_i} = \mathbf{x}^*.$$

Să admitem că punctul $\mathbf{x}^*$ nu este soluţia problemei de programare convexă iniţială şi fie

$$g_{j_h}(\mathbf{x}^*) = 0, \qquad (h = 1, 2, \ldots, h_1), \tag{8.19}$$

$$g_j(\mathbf{x}^*) < 0, \qquad (j \neq j_1, \ldots, j_{h_1})$$

Atunci există o direcţie $\mathbf{s}^*$ de cea mai rapidă scufundare din punctul $\mathbf{x}^*$, care verifică sistemul de inegalităţi

$$\begin{cases} \mathbf{c}\,\mathbf{s}^* \leqslant \eta^* \\ \nabla g_{j_h}(\mathbf{x}^*)\mathbf{s}^* \leqslant \eta^*, \qquad (h = 1, 2, \ldots, h_1) \end{cases} \tag{8.20}$$

unde $\eta^* < 0$.

Din sistemul (8.20) rezultă

$$\begin{cases} \mathbf{c}\,\mathbf{s}^* < \dfrac{\eta^*}{2} \\ \nabla g_{j_h}(\mathbf{x}^*)\mathbf{s}^* < \dfrac{\eta^*}{2} \end{cases} \tag{8.21}$$

Vom arăta acum că în determinarea direcţiei admisibile de scufundare din punctul $\mathbf{x}^{k'_i}$ pentru k'_i suficient de mare pot participa numai funcţiile din sistemul $g_{j_1}, \ldots, g_{j_{h_1}}$ (dar nu neapărat toate).

Într-adevăr, în punctul $\mathbf{x}^*$ sînt îndeplinite relaţiile (8.19). Fie

$$\min_{j \neq j_h} |g_j(\mathbf{x}^*)| = \sigma$$

Atunci

$$g_j(\mathbf{x}^*) < -\frac{\sigma}{2} \qquad (j \neq j_1 \ldots, j_{h_1}).$$

Scanned with OKEN Scanner

De aceea pentru orice punct dintr-o vecinătate oarecare $V_{x^*} \cap \Omega$ a punctului $\mathbf{x}^*$ avem:

$$g_j(\mathbf{x}) < -\frac{\sigma}{2}, \qquad (j \neq j_1, \ldots, j_{h_1}).$$

Din convergența către zero a șirului (δ_k) rezultă acum că pentru k suficient de mare vom avea

$$\delta_k < \frac{\sigma}{2}$$

și de aceea în punctele mulțimii $V_{x^*} \cap \Omega$, și prin urmare și în punctele $\mathbf{x}^{k_i'}$ cu k_i' suficient de mare, în determinarea direcției admisibile de scufundare, nu pot participa funcțiile diferite de $g_{j_1}, g_{j_2}, \ldots, g_{j_{h_1}}$.

Din continuitatea derivatelor parțiale a funcțiilor g_j și din inegalitatea (8.21) rezultă că pentru k_i' suficient de mare, vom avea:

$$\mathbf{c}\,\mathbf{s}^* < \frac{\eta^*}{2}$$

$$\nabla g_{j_h}\left(\mathbf{x}^{k_i'}\right)\mathbf{s}^* < \frac{\eta^*}{2}.$$

Prin urmare, $\mathbf{s}^*$ este de asemenea o direcție admisibilă (nu neapărat de cea mai rapidă scufundare) din punctul $\mathbf{x}^{k_i'}$. Pentru direcția de cea mai rapidă scufundare are loc inegalitatea

$$\eta_{k'} \leqslant \frac{\eta^*}{2} < 0,$$

care contrazice faptul că $\eta_{k_i'} \to 0$.

În felul acesta, presupunînd că $\mathbf{x}^*$ nu este punct optimal pentru problema de programare convexă inițială, am fost conduși la o contradicție. Această contradicție arată că șirul $(f(\mathbf{x}^k))$ este un șir convergent către $f(\mathbf{x}^*)$, deci către cea mai mică valoare a funcției f în domeniul Ω.

2. Varianta a doua a algoritmului Zoutendijk — Zuhoviţki

A doua variantă a algoritmului lui *Zoutendijk-Zuhoviţki* se deosebește de primul doar prin determinarea direcției admisibile de scufundare $\mathbf{s}$. În determinarea direcției admisibile de scufundare din cadrul variantei

Scanned with OKEN Scanner

Scanned with OKEN Scanner

întîi au jucat același rol atît funcția de scop f cît și funcțiile de restricție g_j, $j = 1, 2, \ldots, m$, ceea ce nu este întru totul natural, deoarece scopul nostru principal este să determinăm o direcție admisibilă pe care să descrească cît mai rapid posibil funcția de scop $f(\mathbf{x}) = \mathbf{cx}$ în condițiile $g_j(\mathbf{x}) \leqslant 0$, $j = 1, 2, \ldots, m$. De aceea este mai natural să se determine direcția admisibilă în felul următor:

După găsirea punctului de pornire $\mathbf{x}^0$, alegem odată cu $\delta_1 > 0$ și un număr $\lambda_1 > 0$ suficient de mic. Apoi determinăm direcția de scufundare cu următoarea problemă de programare liniară: *să se minimizeze forma liniară*

$$u = \mathbf{c}\,\mathbf{s} \tag{8.22}$$

în condițiile

$$\begin{cases} \nabla g_j(\mathbf{x}^0)\mathbf{s} \leqslant -\lambda_1, & (j = 1, 2, \ldots, p_1) \\ |s_i| \leqslant C, & (i = 1, 2, \ldots, n). \end{cases} \tag{8.23}$$

Dacă

$$\min u = \eta_1 < -\delta_1,$$

atunci deplasîndu-ne după direcția $\mathbf{s}^0$, vom găsi punctul $\mathbf{x}^1$ ca mai înainte

$$\mathbf{x}^1 = \mathbf{x}^0 + t_1\mathbf{s}^0,$$

pe care-l considerăm apoi ca punct de pornire, punînd

$$\delta_2 = \delta_1 \qquad \lambda_2 = \lambda_1.$$

Continuăm astfel pînă ce ajungem la pasul al k-lea în situația în care

$$\min u = \eta_k \geqslant -\delta_k.$$

În cazul $0 > \eta_k \geqslant -\delta_k$ determinăm t_k, punctul $\mathbf{x}^{k+1}$ și continuăm procesul algoritmic, considerînd

$$\delta_{k+1} = \frac{\delta_k}{2} \quad \text{și} \quad \lambda_{k+1} = \frac{\lambda_k}{2}.$$

Dacă $\eta_k \geqslant 0$, atunci punem $\delta_{k+1} = \frac{\delta_k}{2}$, iar alegerea parametrului λ_k se face în felul următor: Fie

$$g_{j_h}(\mathbf{x}^{k-1}) = 0, \qquad (h = 1, 2, \ldots, h_1);$$

$$g_j(x^{k-1}) < 0, \qquad (j \neq j_1, \ldots, j_{h_1}).$$

Atunci *minimizăm funcţia*

$$u = \mathbf{c}\,\mathbf{s}$$

în condiţiile

$$\nabla g_{j_h}(\mathbf{x}^{k-1})\mathbf{s} \leqslant 0, \qquad (h = 1, 2, \ldots, h_1)$$

$$|s_i| \leqslant C, \qquad (i = 1, 2, \ldots, n).$$

Dacă

$$\min u = \eta_k' = 0,$$

atunci $\mathbf{x}^{k-1}$ este soluţia problemei iniţiale şi procesul algoritmic se termină.

Dacă însă $\eta_k' < 0$, atunci se rezolvă încă o problemă de programare liniară: *să se maximizeze funcţia liniară*

$$v = \lambda$$

în condiţiile

$$\mathbf{c}\,\mathbf{s} - \frac{\eta_k'}{2} \leqslant 0,$$

$$\nabla g_{j_h}(\mathbf{x}^{k-1})\mathbf{s} \leqslant -\lambda \qquad (h = 1, 2, \ldots, h_1);$$

$$|s_i| \leqslant C, \qquad (i = 1, 2, \ldots, n).$$

Dacă $\max v = v_k$, atunci punem

$$\lambda_{k+1} = \min\left\{\frac{\lambda_k}{2}, v_k\right\}$$

Scanned with OKEN Scanner

și continuăm procesul deplasîndu-ne pe direcția s^k, găsită, cu parametrii λ_{k+1} și δ_{k+1} astfel calculați.

Se poate arăta că și acest algoritm este convergent și conduce la soluția problemei inițiale.

Pentru ilustrarea primei variante a algoritmului considerăm următorul exemplu.

Exemplul 8.1: Să se minimizeze funcția

$$f(\mathbf{x}) = x_3$$

în condițiile

$$g_1(\mathbf{x}) = x_1^2 + 2x_1x_2 + 2x_2^2 - 2x_1 - x_2 - x_3 - 2 \leqslant 0,$$

$$g_2(\mathbf{x}) = x_1^2 + x_2^2 - x_1 + x_2 - x_3 - 3 \leqslant 0$$

$$g_3(\mathbf{x}) = x_1^2 + x_1 - 4x_2 - x_3 + 3 \leqslant 0$$

Se poate verifica ușor că cele trei funcții g_1, g_2 și g_3 sînt convexe și deci se poate aplica algoritmul descris.

Ca punct inițial, luăm $\mathbf{x}^0 = (1, -1, 9)$ în care avem

$$g_1(\mathbf{x}^0) = -11, \; g_2(\mathbf{x}^0) = -12, \; g_3(\mathbf{x}^0) = 0.$$

Punem $\delta_1 = 0{,}5$. Atunci

$$g_1(\mathbf{x}^0) = -11 < -\delta_1, \; g_2(\mathbf{x}^0) = -12 < \delta_1$$

$$-\delta_1 < g_3(\mathbf{x}^0) = 0.$$

Pasul 1. 1). Determinarea direcției s^0. Avem:

$$\nabla g_3(\mathbf{x}^0) = (3, -4, -1).$$

Direcția s^0 va fi soluția următoarei probleme de programare liniară

$$u = \eta \to \min$$

în condițiile

$$s_3 - \eta \leqslant 0$$

$$3s_1 - 4s_2 - s_3 - \eta \leqslant 0$$

$$s_1 - 1 \leqslant 0, \; -s_1 - 1 \leqslant 0, \; s_2 - 1 \leqslant 0, \; -s_2 - 1 \leqslant 0.$$

Scanned with OKEN Scanner

S-a omis restricția $|s_3| \leqslant 1$, deoarece s_3 este evident mărginit, ca urmare a primelor două inegalități.

Rezolvînd această problemă găsim soluția

$$\mathbf{s}^0 = (-1;\ 1;\ -3,\ 5),\ \eta_1 = -3,\ 5.$$

2) *Calculul lui* δ_2. Avem $\eta_1 = -3{,}5 < -\delta_1$, astfel că

$$\delta_2 = \delta_1 = 0{,}5.$$

3) *Determinarea lungimii pasului t.* Cea mai mică rădăcină pozitivă a ecuațiilor

$$g_1(\mathbf{x}^0 + t\mathbf{s}^0) = 0,\ g_2(\mathbf{x}^0 + t\mathbf{s}^0) = 0,\ g_3(\mathbf{x}^0 + t\mathbf{s}^0) = 0$$

este $t_1 = 2{,}1$.

4) *Determinarea noului punct de aproximație.*

Avem

$$\mathbf{x}^1 = \mathbf{x}^0 + t_1\mathbf{s}^0 = (-1{,}1;\ 1{,}1;\ 1{,}65).$$

Deci

$$g_1(\mathbf{x}^1) = -1{,}34 < -\delta_2;\ -\delta_2 < g_2(\mathbf{x}^1) = 0.$$

$$g_3(\mathbf{x}^1) = -2{,}94 < -\delta_2.$$

Pasul 2. 1) *Determinarea direcției* $\mathbf{s}^1$. Deoarece

$$\nabla g_2(\mathbf{x}^1) = (-3{,}2;\ 3{,}2;\ -1)$$

se rezolvă problema:

$$u = \eta \to \min$$

în condițiile

$$s_3 - \eta \leqslant 0$$

$$-3{,}2s_1 + 3{,}2s_2 - s_3 - \eta \leqslant 0$$

$$|s_i| \leqslant 1, \qquad (i = 1,\ 2,\ 3).$$

Scanned with OKEN Scanner

Soluţia acestei probleme este

$$\mathbf{s}^1 = (1\,;\ -1\,;\ -3{,}2),\ \eta_2 = -3{,}2.$$

2) *Determinarea lui* δ_3. Avem

$$\eta_2 = -3{,}2 < -\delta_2,$$

de aceea

$$\delta_3 = \delta_2 = 0{,}5.$$

3) *Determinarea pasului* t. Cea mai mică rădăcină a ecuaţiilor

$$g_i(\mathbf{x}^1 + t\mathbf{s}^1) = 0, \qquad (i = 1,\ 2,\ 3),$$

este $t_2 = 0{,}45$.

4) *Determinarea noului punct de aproximaţie.* Avem:

$$\mathbf{x}^2 = \mathbf{x}^1 + t_2\mathbf{s}^1 = (-0{,}65\,;\ 0{,}65\,;\ 0{,}21)$$

Pasul 3. Procedînd la fel se determină succesiv

$$\mathbf{s}^2 = (1\,;\ 1\,;\ -2{,}15),\ \delta_4 = \delta_3 = 0{,}5,\ t_3 = 0{,}36$$

care conduce la noul punct

$$\mathbf{x}^3 = (-0{,}29\,;\ 1{,}01\,;\ -0{,}57).$$

Pasul 4. În mod analog aici se determină succesiv:

$$\mathbf{s}^3 = (1\,;\ -1\,;\ -1{,}51),\ \delta_5 = \delta_4 = 0{,}5,\ t_4 = 0{,}12,$$

$$\mathbf{x}^4 = (-0{,}17\,;\ 0{,}89\,;\ -0{,}75).$$

Pasul 5. Întrucît la acest pas apare o modificare, vom descrie etapele. Astfel avem

$$g_1(\mathbf{x}^4) = -0{,}49,\ g_2(\mathbf{x}^4) = -0{,}37,\ g_3(\mathbf{x}^4) = 0$$

$$\nabla g_1(\mathbf{x}^4) = (0{,}56\,;\ 2{,}22\,;\ -1)$$

Scanned with OKEN Scanner

$$\nabla g_2(\mathbf{x}^4) = (-1{,}34\,;\ 2{,}78\,;\ -1)$$

$$\nabla g_3(\mathbf{x}^4) = (0{,}66\,;\ -4\,;\ -1).$$

1) *Determinarea direcției* $\mathbf{s}^4$. Deoarece

$$-\delta_4 < g_1(\mathbf{x}^4) < 0,\ -\delta_4 < g_2(\mathbf{x}^4) < 0,\ -\delta_4 < g_3(\mathbf{x}^4) = 0$$

direcția de scufundare se determină rezolvînd problema:

$$\min u = \min \eta$$

în condițiile

$$s_3 - \eta \leqslant 0$$

$$0{,}56s_1 + 2{,}22s_2 - s_3 - \eta \leqslant 0$$

$$-1{,}34s_1 + 2{,}78s_2 - s_3 - \eta \leqslant 0$$

$$0{,}66s_1 - 4s_2 - s_3 - \eta \leqslant 0$$

$$|s_i| \leqslant 1, \qquad (i = 1,\ 2,\ 3)$$

Soluția acestei probleme este $\eta_5 = 0$.

2) *Determinarea lui* δ_6. Deoarece avem $\eta_5 = 0$, se rezolvă următoarea problemă de programare liniară

$$\min u = \min \eta$$

în condițiile

$$s_3 - \eta \leqslant 0$$

$$0{,}66s_1 - 4s_2 - s_3 - \eta \leqslant 0$$

$$|s_i| \leqslant 1, \qquad (i = 1,\ 2,\ 3),$$

Soluția acestei probleme este

$$\mathbf{s}^4 = (-1\,;\ 1\,;\ -2{,}33),$$

iar

$$\delta_6 = \frac{\delta_5}{2} = 0{,}25.$$

Scanned with OKEN Scanner

3) *Determinarea pasului t.* Cea mai mică rădăcină pozitivă a ecuațiilor

$$g_i(\mathbf{x}^4 + t\mathbf{s}^4) = 0, \qquad (i = 1, 2, 3)$$

este $t_5 = 0{,}06$.

4) *Determinarea noului punct de aproximație.* Avem

$$\mathbf{x}^5 = \mathbf{x}^4 + t_5\mathbf{s}^4 = (-0{,}23\,;\ 0{,}95\,;\ -0{,}88).$$

În acest punct avem:

$$-\delta_6 < g_1(\mathbf{x}^5) = -0{,}19 < 0,$$

$$-\delta_6 < g_2(\mathbf{x}^5) = 0, \quad -\delta_6 < g_3(\mathbf{x}^5) = -0{,}1 < 0.$$

Pasul 6. 1) *Determinarea direcției* $\mathbf{s}^5$. Se realizează prin rezolvarea problemei de programare liniară

$$\min u = \min \eta$$

în condițiile

$$s_3 - \eta \leqslant 0$$

$$-0{,}56s_1 + 2{,}34s_2 - s_3 - \eta \leqslant 0$$

$$-1{,}46s_1 + 2{,}90s_2 - s_3 - \eta \leqslant 0$$

$$0{,}54s_1 - 4s_2 - s_3 - \eta \leqslant 0$$

$$|s_i| \leqslant 1, \qquad (i = 1, 2, 3).$$

Soluția acestei probleme este

$$\mathbf{s}^5 = (1\,;\ 0{,}18\,;\ -0,\ 08), \quad \eta_6 = -0{,}08.$$

2) *Determinarea lui* δ_7. Deoarece $\eta_6 > \delta_6$, punem

$$\delta_7 = \frac{\delta_6}{2} = 0{,}12.$$

Scanned with OKEN Scanner

3) *Determinarea pasului t.* Rezolvînd ecuaţiile

$$g_i(\mathbf{x}^5 + ts^5) = 0, \qquad (i = 1, 2, 3),$$

găsim că cea mai mică rădăcină pozitivă este $t_6 = 0{,}34$.

4) *Determinarea noului punct de aproximaţie.* Avem

$$\mathbf{x}^6 = \mathbf{x}^5 + t_6 s^5 = (0{,}11\,;\ 1{,}01\,;\ -0{,}91)$$

şi

$$-\delta_7 < g_1(\mathbf{x}^6) = -0{,}05 < 0,\ g_2(\mathbf{x}^6) = -0{,}16 < -\delta_7$$

$$-\delta_7 < g_3(\mathbf{x}^6) = 0.$$

Pasul 7. În urma efectuării acestui pas se găseşte

$$s^6 = (-1\,;\ -0{,}15\,;\ -0{,}26),\ \delta_8 = \delta_7 = 0{,}12,$$

$$t_7 = 0{,}2$$

iar

$$\mathbf{x}^7 = (-0{,}09\,;\ 0{,}98\,;\ -0{,}96).$$

Pasul 8. În urma acestui pas se obţine

$$s^7 = (0{,}3\,;\ -0{,}03\,;\ -0{,}18),\ \delta_9 = \delta_8 = 0{,}12,\ t_8 = 0{,}09$$

iar

$$\mathbf{x}^8 = (-0{,}06\,;\ 0{,}98\,;\ -0{,}98).$$

Urmărind şirul aproximaţiilor succesive $\mathbf{x}^0, \mathbf{x}^1, \cdots, \mathbf{x}^8$, se vede că punctul $\mathbf{x}^* = (0,\ 1,\ -1)$ trebuie să fie soluţia optimă căutată.

Pentru a ne convinge de aceasta vom aplica criteriul de optim. Astfel, avem:

$$g_1(\mathbf{x}^*) = g_2(\mathbf{x}^*) = g_3(\mathbf{x}^*) = 0.$$

$$\nabla g_1(\mathbf{x}^*) = (0,\ 3,\ -1),\ \nabla g_2(\mathbf{x}^*) = (-1\,;\ 3\,;\ -1)$$

$$\nabla g_3(\mathbf{x}^*) = (1,\ -4,\ -1).$$

Scanned with OKEN Scanner

În rezolvarea problemei de programare liniară

$$\min u = \min \eta$$

în condiţiile

$$s_3 - \eta \leqslant 0$$

$$3s_2 - s_3 - \eta \leqslant 0$$

$$-s_1 + 3s_2 - s_3 - \eta \leqslant 0$$

$$s_1 - 4s_2 - s_3 - \eta \leqslant 0$$

$$|s_1| \leqslant 1, \ |s_2| \leqslant 1$$

se găseşte $\min u = \eta^* = 0$ şi de aceea pe baza criteriului de optim rezultă că punctul $\mathbf{x}^*$ este punctul de optim căutat.

§ 9. Metoda gradientului redus

Vom prezenta acum o metodă relativ recentă, dată de WOLFE, PH. [79] în anul 1967, pentru o programare concavă cu restricţii liniare, numită metoda *gradientului redus*.

Metoda este dată pentru următoarea problemă de programare concavă: *să se maximizeze funcţia concavă*

(9.1) $$f(\mathbf{x})$$

în condiţiile

(9.2) $$\mathbf{A}\mathbf{x} \leqslant \mathbf{b},$$

unde $\mathbf{A}$ *este o matrice de tipul* (m, n).

Se presupune că $m \geqslant n$ şi că orice submatrice a matricei $\mathbf{A}$ de tipul (p, n) are rangul p, pentru $p \leqslant n$. Atunci problema considerată *este nedegenerată*. În particular, orice matrice pătratică de tipul (n, n) are rangul egal cu n. În acest caz matricea $\mathbf{A}$ se descompune într-o matrice $\mathbf{C}$ de tipul (n, n) şi o matrice $\mathbf{D}$ de tipul $((m - n), n)$. Această descompunere a matricei $\mathbf{A}$ implică o descompunere a vectorului $\mathbf{b}$ respectiv în vectorul $\mathbf{c}$ şi $\mathbf{d}$.

Inegalităţile (9.2) se pot scrie atunci astfel:

$$\mathbf{C}\mathbf{x} \leqslant \mathbf{c}$$

$$\mathbf{D}\mathbf{x} \leqslant \mathbf{d}.$$

Scanned with OKEN Scanner

Introducînd acum n variabile auxiliare $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ şi $m-n$ variabile $z = (z_1, z_2, \ldots, z_{m-n})$, vom obţine noul sistem de restricţii sub forma:

$$\mathbf{Cx} + \mathbf{y} = \mathbf{c} \tag{9.3}$$

$$\mathbf{Dx} + \mathbf{z} = \mathbf{d},\ \mathbf{y},\ \mathbf{z} \geqslant \mathbf{0}. \tag{9.4}$$

Din relaţia (9.3) rezultă

$$\mathbf{x} = \mathbf{C}^{-1}\mathbf{c} - \mathbf{C}^{-1}\mathbf{y}, \tag{9.5}$$

care înlocuită în inegalitatea (9.4), conduce la un sistem de restricţii de forma

$$\mathbf{z} = \overline{\mathbf{b}} - \overline{\mathbf{A}}\mathbf{y}, \quad \mathbf{y},\ \mathbf{x} \geqslant \mathbf{0}, \tag{9.6}$$

care este echivalent cu sistemul (9.3)—(9.4).

Din ipoteza făcută rezultă că vectorul $\mathbf{y}$ are ca şi componente variabilele independente, iar vectorul $\mathbf{z}$ cele dependente. Vîrfurile domeniului admisibil Ω sînt caracterizate prin anularea a celor n componente a vectorului variabilelor independente $\mathbf{y}$. Prin urmare, pentru a obţine un vîrf al poliedrului Ω este necesar să se aleagă $\mathbf{y}$ astfel încît fiecare variabilă independentă care se anulează să fie conţinută printre variabilele y. Această alegere este importantă, deoarece atunci orice variaţie, destul de mică, $\Delta\mathbf{y}$, cu $\Delta y_i \geqslant 0$, pentru $y_i = 0$, garantează nenegativitatea variabilelor dependente z.

Procedura metodei este analoagă cu cea care rezultă din metoda simplex. Anume, pornind de la un vector $\mathbf{y}$ al variabilelor independente care se anulează, se trece la un alt vector $\mathbf{y}$ care se obţine prin înlocuirea unei variabile independente cu o variabilă dependentă din $\mathbf{z}$.

Algoritmul poate fi descris în felul următor:

Mai întîi se calculează din (9.3) şi (9.4) creşterile

$$\Delta\mathbf{x} = -\mathbf{C}^{-1}\Delta\mathbf{y} \tag{9.7}$$

$$\Delta\mathbf{z} = \mathbf{D}\mathbf{C}^{-1}\Delta\mathbf{y}$$

şi expresia

$$\nabla_y f(\mathbf{x}) = -\nabla f(\mathbf{x})\,\mathbf{C}^{-1} \tag{9.8}$$

numită *gradientul redus*.

Plecînd apoi de la un punct al domeniului Ω (soluţie admisibilă) $\mathbf{x}^k$, se procedează în continuare conform algoritmului construit cu următorii patru paşi:

Scanned with OKEN Scanner

1°. Se ia

$$\Delta y_i = \begin{cases} [\nabla_y f(\mathbf{x}^k)]_i, & \text{dacă } [\nabla_y f(\mathbf{x}^k)]_i > 0, \text{ sau } y_i > 0, \\ 0, & \text{în rest} \end{cases}$$

unde prin $[\nabla_y f(\mathbf{x})]_i$ s-a notat componenta a i-a a vectorului $\nabla_y f(\mathbf{x})$.

2°. Dacă $\Delta \mathbf{y} = 0$, $\mathbf{x}^k$ este punctul de maxim căutat. În caz contrar se formează din nou $\Delta \mathbf{x}$ şi $\Delta \mathbf{z}$ după formulele (9.7)

3°. Se pune

$$\max \{t | \mathbf{y} + t\Delta \mathbf{y} \geqslant \mathbf{0}\} = t_1$$

$$\max \{t | \mathbf{z} + t\,\Delta \mathbf{z} \geqslant \mathbf{0}\} = t_2$$

$$\max \{f(\mathbf{x}^k + t\Delta \mathbf{x})\} = t_3$$

unde $t \geqslant 0$. Fie

$$\tau = \min (t_1, t_2, t_3).$$

Atunci următorul punct de aproximaţie succesivă este

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \tau\,\Delta \mathbf{x},$$

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \tau\,\Delta \mathbf{y}$$

$$\mathbf{z}^{k+1} = \mathbf{z}^k + \tau\,\Delta \mathbf{z}.$$

4°. Dacă $\tau < t_2$, atunci se anulează din nou o variabilă independentă y_i şi se trece din nou la punctul 1°. În caz contrar un z_i se ia ca variabilă independentă, iar cea mai mare variabilă independentă y_i va deveni variabilă dependentă. Se construieşte transformarea pivot (9.7) şi se trece din nou la punctul 1°.

Experienţa de pînă acum, după cum afirmă KÜNZI, H. P. [44], a arătat că această metodă este foarte eficace. De curînd acest lucru a fost justificat cu ocazia rezolvării unei probleme legate de distribuirea energiei electrice de la Trustul „Electricité de France".

Autorul metodei WOLFE, PH. afirmă că s-a rezolvat cu această metodă o problemă de programare concavă cu 200 restricţii şi 1000 de variabile. Metoda a fost programată la o maşină electronică 36-Bit.

Scanned with OKEN Scanner

Un avantaj în plus al acestei metode constă în aceea că utilizează în mare măsură algoritmul simplex, pentru care, după cum am amintit deja, dispunem de programe bine verificate.

Ca dezavantaj menţionăm faptul că se aplică numai în cazul cînd restricţiile sînt liniare.

§ 10. Metoda centrelor a lui Huard

Vom prezenta acum o metodă cunoscută în literatură sub denumirea de *metoda centrelor* elaborată în 1964 de HUARD, P. [29]

Această metodă se referă la o problemă generală de programare concavă cu restricţii convexe. Astfel, se consideră problema: *să se maximizeze funcţia concavă*

$$f(\mathbf{x}) \tag{10.1}$$

în condiţiile

$$g_j(\mathbf{x}) \geqslant 0, \qquad (j = 1, 2, \ldots, m), \tag{10.2}$$

unde $g_j(\mathbf{x})$ sînt de asemenea funcţii concave.

Şi aici se presupune că domeniul soluţiilor Ω este mărginit şi are interiorul nevid. Se presupune de asemenea că printre restricţii nu figurează şi egalităţi. De altfel se ştie că o egalitate se poate înlocui cu două inegalităţi.

Algoritmul pe care-l propune Huard este foarte general, deoarece depinde de o *funcţie-distanţă* de la un punct interior al domeniului Ω la frontiera sa. De aceea alegînd funcţia-distanţă în diverse moduri se pot obţine diverse variante ale algoritmului.

Independent de distanţa considerată, algoritmul are următoarele caracteristici:

1. Fiind o metodă iterativă nu conduce la acumularea de erori.
2. Şirul aproximaţiilor succesive este un şir de puncte în care şirul valorilor funcţiei de scop $f(\mathbf{x})$ este un şir strict crescător.
3. Paşii sînt extrem de simpli şi nu sînt influenţaţi de particularităţile frontierei domeniului Ω (ca vîrfuri, puncte extreme degenerate etc.).

Vom trece acum la descrierea metodei. În prealabil vom da noţiunea de *distanţă*.

Se numeşte funcţie — distanţă a punctului $\mathbf{x}$ *al domeniului* Ω *la frontiera lui* Ω, o funcţie $\theta(\mathbf{x};\Omega)$ continuă pe Ω şi care are următoarele patru proprietăţi:

a) $\theta(\mathbf{x};\ \Omega) = 0$ pentru orice punct $\mathbf{x}$ al frontierei domeniului Ω;

b) $\theta(\mathbf{x};\ \Omega) > 0$ pentru orice punct $\mathbf{x}$ din interiorul domeniului Ω.

c) Dacă Ω' este un domeniu care se obţine din Ω prin deplasarea unei restricţii, astfel încît $\Omega' \subset \Omega$, atunci

$$\theta(\mathbf{x};\Omega') \leqslant \theta(\mathbf{x};\Omega), \quad \forall\, \mathbf{x} \in \Omega'$$

Scanned with OKEN Scanner

d) Fie $g_j(\mathbf{x}; \lambda) \geqslant 0$ o restricţie considerată în *c*), λ fiind un parametru care determină poziţia restricţiei, şi fie Ω'_λ domeniul corespunzător. Dacă $\mathbf{x}(\lambda)$ este un punct variabil, care depinde de acelaşi parametru λ şi care aparţine domeniului Ω'_λ, atunci din

$$g_j[\mathbf{x}(\lambda); \lambda] \to 0, \quad \lambda \to \infty,$$

rezultă

$$\theta[\mathbf{x}(\lambda), \Omega'_\lambda] \to 0, \qquad \lambda \to \infty.$$

Vom introduce acum noţiunea fundamentală de *centru.* Fiind dat un domeniu convex Ω şi o funcţie —distanţă $\theta(\mathbf{x}; \Omega)$, definită pe acest domeniu, punctul $\tilde{\mathbf{x}} \in \Omega$ se numeşte *centru al domeniului* Ω, dacă punctul $\tilde{\mathbf{x}}$ maximizează funcţia distanţă pe Ω, adică

$$\theta(\tilde{\mathbf{x}}; \Omega) = \max \{\theta(\mathbf{x}; \Omega) | \mathbf{x} \in \Omega\}. \tag{10.3}$$

Descrierea algoritmului. Algoritmul porneşte de la un punct arbitrar $\mathbf{x}^0$ al domeniului. Se consideră apoi domeniul convex R_1 definit în felul următor:

$$R_1 = \Omega \cap \{\mathbf{x} | f(\mathbf{x}) \geqslant f(\mathbf{x}^0)\}.$$

Fie acum $\mathbf{x}^1$ centrul domeniului R_1, adică

$$\theta(\mathbf{x}^1; R_1) = \max \{\theta(\mathbf{x}; R_1) | \mathbf{x} \in R_1\}.$$

Cu punctul $\mathbf{x}^1$ se procedează la fel ca şi cu punctul $\mathbf{x}^0$, şi procesul continuă (Figura 10.1).

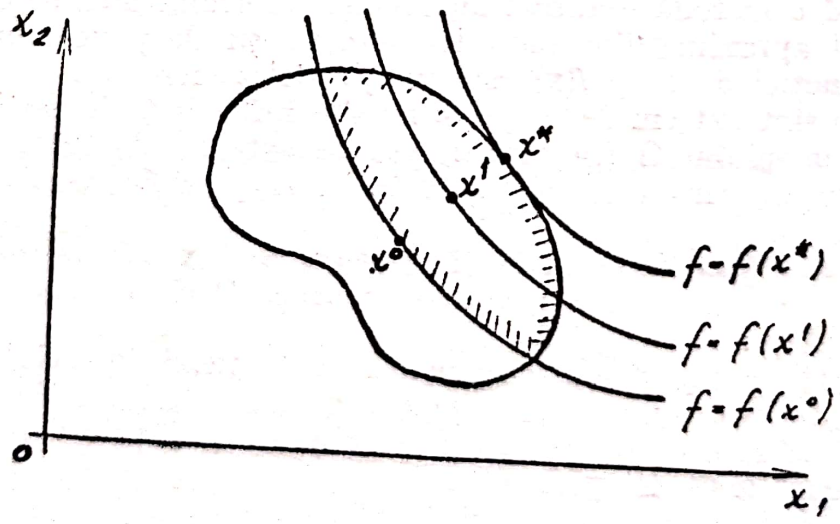

Figura 10.1.

Scanned with OKEN Scanner

Autorul metodei demonstrează că şirul centrelor $(\mathbf{x}^k)$ astfel obţinut tinde către punctul optimal. În general soluţia se obţine după un număr infinit de paşi. Soluţia poate fi atinsă după un număr finit de paşi numai în cazul cînd funcţia $f(\mathbf{x})$ îşi *atinge maximul în interiorul lui* Ω.

După cum se vede, algoritmul depinde în mod esenţial de alegerea funcţiei-distanţă. Aceasta din urmă trebuie aleasă destul de simplă, pentru ca, calculul centrului să fie suficient de simplu. Autorul propune două funcţii-distanţă :

$$\theta(\mathbf{x};\ R_k) = [f(\mathbf{x}) - f(\mathbf{x}^k)] \prod_{j=1}^{m} g_j(\mathbf{x}) \tag{10.4}$$

$$\theta(\mathbf{x};\ R_k) = \min \{f(\mathbf{x}) - f(\mathbf{x}^k),\ g_j(\mathbf{x}) | j = 1, 2, \ldots, m\}. \tag{10.5}$$

Funcţia-distanţă, definită în (10.4) *nu este concavă*, însă se demonstrează că aceasta nu posedă maxim local care să nu fie punct optimal. Aceasta permite folosirea metodei gradientului pentru găsirea maximului ei în domeniul Ω. A doua funcţie este concavă, dar nu este comodă pentru practică. Funcţia din (10.4) are o proprietate interesantă din punct de vedere al calculului. În particular direcţia gradientului său rămîne invariantă cînd se multiplică $g_j(\mathbf{x})$ cu coeficienţi arbitrari, care permite considerarea unui număr mare de restricţii şi totodată domenii R_k foarte mici.

În sfîrşit se observă că funcţia (10.4) este analoagă cu funcţia potenţial logaritmic din cunoscuta metodă a lui FRISCH R.A.K. [20]. Comportarea sa în procesul calculelor este complet diferită în cele două metode.

În cazul cînd restricţiile sînt liniare, atunci calculul maximului funcţiei-distanţă devine ea însăşi o problemă de programare, cu restricţii liniare şi funcţia de scop concavă.

Unul din avantajele acestei metode constă în faptul că poate fi aplicată oricărei probleme de programare concavă în care variabilele sînt constrînse într-un domeniu convex arbitrar (nu neapărat definit sub forma unor inegalităţi).

Convergenţa algoritmului

Vom prezenta aici cîteva proprietăţi mai importante ale funcţiei-distanţă cu ajutorul cărora se va putea demonstra apoi şi convergenţa algoritmului descris mai sus. Vom introduce mai întîi cîteva notaţii. Astfel scriind vectorial sistemul de inegalităţi (10.2) sub forma $\mathbf{g}(\mathbf{x}) \geqslant 0$, notăm :

$$\begin{aligned}
R(\lambda) &= \{\mathbf{x} | \mathbf{g}(\mathbf{x}) \geqslant 0,\ \ f(\mathbf{x}) \geqslant \lambda\} \\
S(\lambda) &= \{\mathbf{x} | \mathbf{g}(\mathbf{x}) \geqslant 0,\ \ f(\mathbf{x}) > \lambda\} \\
F(\lambda) &= \{\mathbf{x} | \mathbf{x} \in R(\lambda), \exists j_0 \in \{1, 2, \ldots, m\}, g_{j_0}(\mathbf{x}) = 0 \text{ sau } f(\mathbf{x}) = \lambda\} \\
I(\lambda) &= \{\mathbf{x} | g_j(\mathbf{x}) > 0,\ \ \forall j \in \{1, 2, \ldots, m\},\ \ f(\mathbf{x}) > \lambda\}.
\end{aligned} \tag{10.6}$$

Scanned with OKEN Scanner

Se vede că domeniile $F(\lambda)$ și $I(\lambda)$ sînt respectiv frontiera și interiorul lui $R(\lambda)$. În ce privește domeniul $R(\lambda)$ aceasta reprezintă intersecția dintre Ω și $\{\mathbf{x}|f(\mathbf{x}) \geqslant \lambda\}$. Aici λ este un parametru.

Funcția distanță $\theta(\mathbf{x}, \lambda) = \theta(\mathbf{x}, R(\lambda))$, fiind continuă în raport cu $\mathbf{x}$, ea rămîne continuă pentru $\mathbf{x} \in R(\lambda)$, oricare ar fi λ astfel ca $R(\lambda) \neq \emptyset$.

Se fac următoarele ipoteze relativ la domeniul inițial și funcția distanță:

H_0. Domeniul Ω este compact.

H_1. Domeniul Ω conține puncte interioare, adică există cel puțin un punct $\mathbf{x}^0 \in \mathbf{R}^n$ astfel ca $\mathbf{g}(\mathbf{x}^0) > 0$.

H_2. $\theta(\mathbf{x}, \lambda) = 0$ dacă $\mathbf{x} \in F(\lambda)$.

H_3. $\theta(\mathbf{x}, \lambda) > 0$ dacă $\mathbf{x} \in I(\lambda)$.

H_4. Dacă $\lambda' > \lambda \Rightarrow \theta(\mathbf{x}, \lambda') \leqslant \theta(\mathbf{x}, \lambda)$, $\forall\, \mathbf{x} \in R(\lambda')$, adică $\theta(\mathbf{x}, \lambda)$ este o funcție descrescătoare în raport cu parametrul λ.

H_5. $f[\mathbf{x}(\lambda)] - \lambda \to 0$ implică $\theta[\mathbf{x}(\lambda), \lambda] \to 0$, unde prin $\mathbf{x}(\lambda)$ se înțelege un punct din $R(\lambda)$.

TEOREMA 10.1. 1). *Dacă* $S(\lambda) \neq \emptyset$ *atunci și* $I(\lambda) \neq \emptyset$. 2) *Dacă* $I(\lambda) = \emptyset$, *atunci* $S(\lambda) = \emptyset$.

Demonstrație. 1) Într-adevăr, dacă $S(\lambda) \neq \emptyset$, atunci pe baza ipotezei H_1, există un punct $\mathbf{x}^0 \in \Omega$ și $\mathbf{y}^0 \in S(\lambda)$ astfel că

$$\mathbf{g}(\mathbf{x}^0) > 0 \tag{10.7}$$

$$\mathbf{g}(\mathbf{y}^0) \geqslant 0,\ f(\mathbf{y}^0) > \lambda. \tag{10.8}$$

Putem presupune că $\mathbf{x}^0 \neq \mathbf{y}^0$, căci altfel teorema este evidentă. Fie $t \in [0,1]$ și

$$\mathbf{x}(t) = t\mathbf{x}^0 + (1 - t)\,\mathbf{y}^0.$$

Din cauza concavității lui $\mathbf{g}(\mathbf{x})$, avem

$$\mathbf{g}[\mathbf{x}(t)] \geqslant t\mathbf{g}(\mathbf{x}^0) + (1 - t)\,\mathbf{g}(\mathbf{y}^0),\ \forall\, t \in [0,\ 1].$$

Din (10.7) și (10.8) rezultă

$$\mathbf{g}[\mathbf{x}(t)] > 0,\ \forall\, t \in\ ]0,\ 1]. \tag{10.9}$$

Din concavitatea lui f avem

$$f[\mathbf{x}(t)] \geqslant tf(\mathbf{x}^0) + (1 - t)\,f(\mathbf{y}^0) = t[f(\mathbf{x}^0) - f(\mathbf{y}^0)] + f(\mathbf{y}^0).$$

Scanned with OKEN Scanner

Însă din (10. 8) rezultă atunci $f[\mathbf{x}(t)] > \lambda$ oricare ar fi $t \in [0, \bar{t}[$, unde

$$\bar{t} = [f(\mathbf{y}^0) - \lambda] / |f(\mathbf{x}^0) - f(\mathbf{y}^0)| > 0.$$

De aici și (10.9) rezultă că există $t^0 \in]\,0,\ 1]$ astfel încît:

$$\mathbf{g}[\mathbf{x}(t^0)] > 0$$

și

$$f[\mathbf{x}(t^0)] > \lambda,$$

care arată că $R(\lambda)$ are interiorul $I(\lambda)$ nevid.

Afirmația 2) este logic echivalentă cu 1).

Notăm:

$$\theta_k(\mathbf{x}) = \theta[\mathbf{x},\ f(\mathbf{x}^k)].$$
$$S_k = S[f(\mathbf{x}^k)]$$
$$R_k = R[f(\mathbf{x}^k)]$$

și fie $f^* = f(\mathbf{x}^*)$, $\mathbf{x}^*$ fiind soluția optimă a problemei inițiale.

Cu aceste notații, algoritmul lui Huard se descrie astfel:

Dacă $S_k = \emptyset$, atunci $\mathbf{x}^k = \mathbf{x}^*$, adică $\mathbf{x}^k$ este punctul de optim căutat și algoritmul se încheie.

În caz contrar, se determină iterația următoare $\mathbf{x}^{k+1}$ astfel ca:

$$\theta_k(\mathbf{x}^{k+1}) = \max\{\theta_k(\mathbf{x}) \mid \mathbf{x} \in R_k\}.$$

Rezultă că:

(10.10) $$\mathbf{x}^{k+1} \in I[f(\mathbf{x}^k)] \text{ și } \theta_k(\mathbf{x}^{k+1}) > 0.$$

Din ipoteza H_0 rezultă că R_k este compact și deci maximul este atins, adică $\mathbf{x}^{k+1} \in R_k$.

Mai departe se procedează la fel înlocuind $\mathbf{x}^k$ cu $\mathbf{x}^{k+1}$.

TEOREMA 10.2. *Oricare ar fi k avem:*

$$f(\mathbf{x}^k) < f(\mathbf{x}^{k+1}) \leqslant f^*.$$

Demonstrația acestei teoreme este evidentă căci prima inegalitate rezultă din definiția lui S_k iar a doua din faptul că oricare ar fi k rezultă că $\mathbf{x}^k \in \Omega$.

Scanned with OKEN Scanner

Din Teorema 10.2., rezultă că şirul $(f(\mathbf{x}^k))$ este un şir convergent şi

$$\lim_{k\to\infty} f(\mathbf{x}^k) = \tilde{f} \leqslant f^*. \tag{10. 11}$$

De asemenea, avem:

$$\lim_{k\to 0} [f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)] = 0. \tag{10.12}$$

Însă conform ipotezei H_5, avem

$$f[\mathbf{x}(\lambda)] - \lambda \to 0 \Rightarrow \theta\,[\mathbf{x}(\lambda),\ \lambda] \to 0$$

şi astfel din (10. 12) rezultă

$$\lim_{k\to\infty} \theta_k(\mathbf{x}^{k+1}) = 0. \tag{10. 13}$$

Dacă notăm:

$$\tilde{R} = R(\tilde{f}) = \{\mathbf{x} | \mathbf{g}(\mathbf{x}) \geqslant 0,\ f(\mathbf{x}) \geqslant \tilde{f}\}$$
$$\tilde{S} = S(\tilde{f}) = \{\mathbf{x} |\ \mathbf{g}(\mathbf{x}) \geqslant 0,\ f(\mathbf{x}) > \tilde{f}\}$$

atunci are loc

TEOREMA 10.3 *Domeniul $\tilde{R}$ are interiorul vid.*

Demonstraţie. Într-adevăr, dacă $\operatorname{int}\tilde{R} \neq \emptyset$, atunci există $\mathbf{x}^0$ astfel încît

$$g_j(\mathbf{x}^0) > 0, \qquad (j = 1,\ 2,\ \ldots,\ m)$$

$$f(\mathbf{x}^0) > \tilde{f},$$

şi din ipoteza H_3 rezultă $\theta\,(\mathbf{x}^0, \tilde{f}) > 0$. Însă deoarece $\tilde{R} \subset R_k$ oricare ar fi k, avem $\mathbf{x}^0 \in R_k$ oricare ar fi k. Din definiţia lui $\mathbf{x}^{k+1}$ ca fiind punctul în care se atinge maximul lui $\theta_k(\mathbf{x})$ pe R_k rezultă:

$$\theta_k(\mathbf{x}^0) \leqslant \theta_k(\mathbf{x}^{k+1}),\ \forall k \in \mathbf{N}. \tag{10. 14}$$

În sfîrşit, utilizînd ipoteza H_4, avem

$$\theta_k(\mathbf{x}^0) = \theta[\mathbf{x}^0, f(\mathbf{x}^k)] \geqslant \theta(\mathbf{x}^0, \tilde{f}),\ \forall k \in \mathbf{N}. \tag{10.15}$$

Scanned with OKEN Scanner

Din (5.14) şi (5.15) rezultă

$$\theta_k(x^{k+1}) \geqslant \theta(x^0, \tilde{f}) > 0, \quad \forall k \in \mathbf{N},$$

care contrazice (10.13). Prin urmare $\operatorname{int}\tilde{R} = \emptyset$ şi teorema este demonstrată.

Ca o consecinţă a Teoremelor 10.1 şi 10.3 rezultă acum că

(10.16) $$\tilde{f} = f^*$$

căci din Teoremele 10.1 şi 10.3 rezultă că $\tilde{S} = \emptyset$ şi deoarece $\Omega \neq \emptyset$ obţinem $f^* \leqslant \tilde{f}$. De aici şi din (10.11) rezultă (10.16). Cu aceasta convergenţa algoritmului este complet demonstrată.

§ 11. Metoda SUMT a lui Fiacco şi McCormick

Denumirea metodei provine de la iniţialele denumirii complete a metodei „Sequencial unconstrained minimization technique". Metoda se bazează pe ideea înlocuirii funcţiei de scop iniţiale cu o altă funcţie, care să înglobeze şi restricţiile problemei de programare şi care să permită apoi rezolvarea problemei de maxim sau minim fără restricţii. Funcţia de scop modificată trebuie să fie de aşa manieră încît în Ω să difere puţin de funcţia iniţială iar în exteriorul lui Ω să aibă valori mici în comparaţie cu cele din domeniul Ω.

Astfel, considerînd problema de programare: *să se maximizeze funcţia concavă*

(11.1) $$f(x)$$

în condiţiile

(11.2) $$g_j(x) \leqslant 0 \quad , \qquad (j = 1, 2, \ldots, m),$$

unde g_j sînt funcţii convexe, se ataşează acesteia funcţia auxiliară

(11.3) $$G(x) = f(x) - \rho\, \Phi[g_1(x), \ldots, g_m(x)],$$

unde $\rho > 0$ este o constantă, iar $\Phi(u_1, u_2, \ldots, u_m)$ o funcţie oarecare, mărginită inferior pentru orice $u_i \leqslant 0$ şi este supusă condiţiei că

$$\lim_{g_j \to 0} \Phi(g_1, \ldots, g_m) = +\infty \text{ pentru orice } j = 1, 2, \ldots, m.$$

Scanned with OKEN Scanner

Autorii consideră, de exemplu, funcţia

$$\Phi(g_1, \ldots, g_m) = -\sum_{j=1}^{m} \frac{1}{g_j(\mathbf{x})}$$

care verifică condiţiile de mai sus.

Algoritmul constă în următoarele: se consideră un punct arbitrar $\mathbf{x}^0 \in \text{int}(\Omega) = R_0$. Se rezolvă apoi problema de maxim pentru funcţia (11. 3) pentru un număr arbitrar, dar fix, $\rho = \rho_0 > 0$. Fie $\mathbf{x}^1 \in R_0$ punctul de maxim pentru funcţia $G(\mathbf{x})$. Alegînd apoi un alt număr $\rho_1 < \rho_0$ se repetă procedeul şi se ajunge la un nou punct $\mathbf{x}^2$ ş.a.m.d.

Considerînd un şir de numere pozitive (ρ_k),

$$\rho_0 > \rho_1 > \ldots > \rho_k > \ldots, \quad \rho_k \to 0,$$

se construieşte şirul aproximaţiilor succesive $(\mathbf{x}^k)$ şi se demonstrează [19] că

$$\tilde{\mathbf{x}} = \lim_{k \to \infty} \mathbf{x}^k$$

este tocmai punctul optimal al problemei iniţiale (11.1) — (11.2).

§ 12. Metoda lui Kleibohm

Mai menţionăm în cadrul acestor metode generale de programare neliniară, o metodă elaborată nu de mult (în anul 1966) de către KLEIBOHM, K. [36]. Această metodă se poate încadra în metoda paşilor mari, deoarece se aseamnănă destul de mult cu celelalte variante ale metodei paşilor mari. Ideea procedeului constă în înlocuirea restricţiilor neliniare cu anumite restricţii liniare, care aproximează din ce în ce mai mult domeniul Ω.

Să considerăm deci din nou problema: *să se maximizeze funcţia concavă*:

(12.1) $$f(\mathbf{x})$$

în condiţiile

(12.2) $$g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots; m),$$

unde g_j sînt toate funcţii convexe şi diferenţiabile.

După cum am văzut, se poate totdeauna presupune că funcţia de scop este liniară, adică

(12.3) $$f(\mathbf{x}) = \mathbf{cx}.$$

Scanned with OKEN Scanner

Se presupune și aici că domeniul soluțiilor inegalităților (12.2) Ω este mărginit. Atunci acest domeniu se poate închide într-un poliedru L_0.

Algoritmul constă în următoarele:

Mai întîi se optimizează funcția de scop $f(\mathbf{x})$ pe poliedrul L_0. Fie $\mathbf{x}^0$ soluția problemei. Se alege apoi în Ω un punct arbitrar P_0 care se unește cu punctul $\mathbf{x}^0$. Dreapta construită înțeapă frontierea domeniului Ω într-un punct P_1 (Figura 12.1). În punctul P_1 se duce un hiperplan tangent. Dacă punctul P_1 se găsește pe mai multe suprafețe de restricție, atunci se consideră hiperplanul tangent la una din suprafețele care trec prin punctul P_1.

Acest hiperplan împarte domeniul L_0 în două regiuni. Fie L_1 regiunea care conține domeniul Ω. Cu domeniul poliedral L_1 se procedează apoi la fel, adică se rezolvă problema de maxim pentru funcția (12.3) și se găsește un punct optimal $\mathbf{x}^1$, care se unește tot cu punctul P_0. Dreapta determinată de punctele $\mathbf{x}^1$ și P_0 înțeapă frontiera domeniului Ω într-un punct P_2, în care se duce din nou hiperplanul tangent (sau unul din hiperplanele tangente) la suprafața ce trece prin P_2 ș.a.m.d.

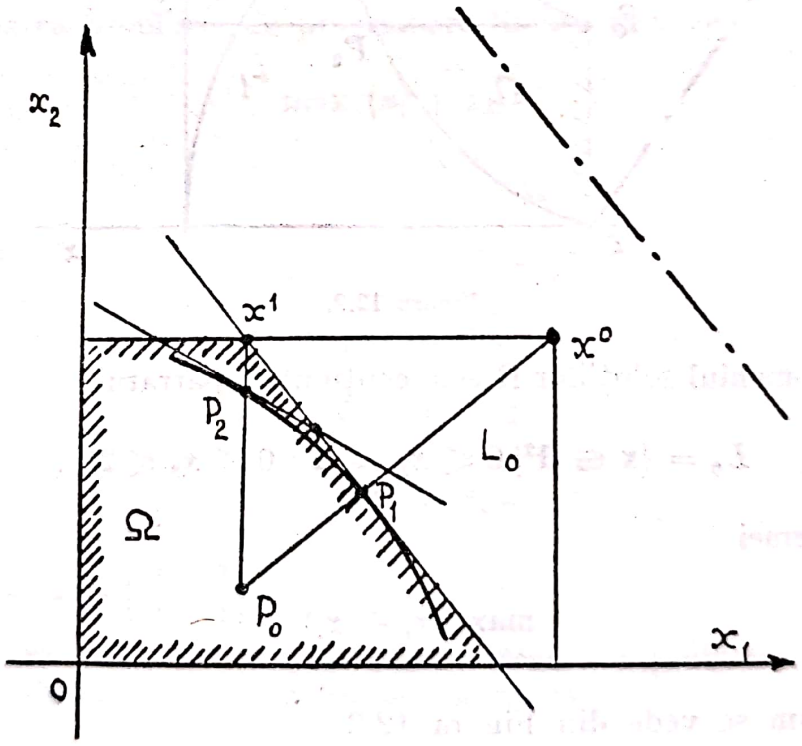

Figura 12.1.

Astfel, se va construi un șir de aproximații succesive $(\mathbf{x}^k)$, care converge, după cum a arătat Kleibohm, [36] către soluția problemei inițiale. În cazul $n = 2$, metoda este ilustrată în Figura 12.1.

Exemplu: Pentru ilustrarea metodei să considerăm exemplul următor

Scanned with OKEN Scanner

$$\max\ f(\mathbf{x}) = \max\ (x_1 + x_2)$$

în condiţiile

$$g_1(\mathbf{x}) = x_1^2 + x_2^2 - 4 \leqslant 0$$

$$g_2(\mathbf{x}) = x_1^2 - x_2 \leqslant 0$$

$$x_1 \geqslant 0,\ x_2 \geqslant 0.$$

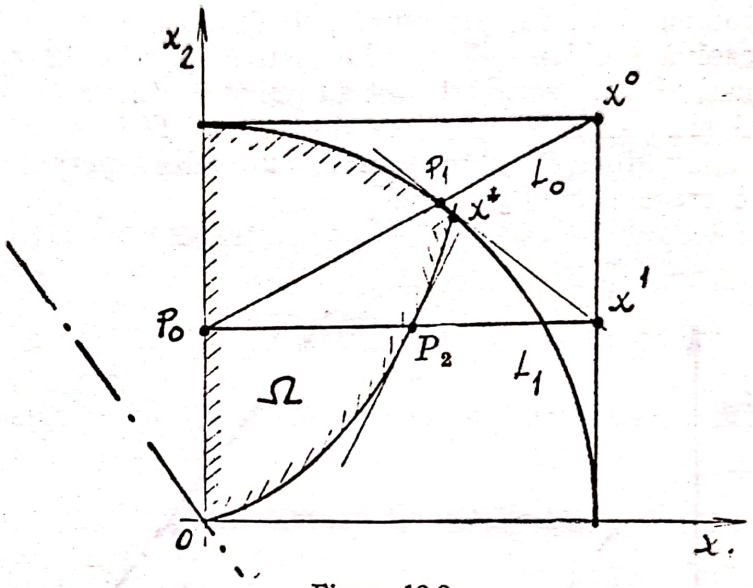

Figura 12.2.

Se vede că domeniul soluţiilor Ω este conţinut în pătratul

$$L_0 = \{\mathbf{x} \in \mathbf{R}^2 |\ 0 \leqslant x_1, \leqslant 2;\ 0 \leqslant x_2 \leqslant 2\}\ .$$

Soluţia problemei

$$\max_{\mathbf{x} \in L_0}\ (x_1 + x_2)$$

este, după cum se vede din Figura 12.2

$$\mathbf{x}^0 = (2,\ 2)$$

Considerînd punctul $P_0 = (0,\ 1)$, dreapta $P_0\mathbf{x}^0$ taie frontiera domeniului Ω în punctul $P_1 = \left(\frac{6}{5},\ \frac{8}{5}\right)$. Ecuaţia tangentei la cercul $x_1^2 + x_2^2 - 4 = 0$ în P_1 este $3x_1 + 4x_2 - 10 = 0$

Prin urmare, următoarea problemă de programare liniară va fi:

$$\max\ (x_1 + x_2)$$

în condiţiile

$$3x_1 + 4x_2 \leqslant 10$$

$$0 \leqslant x_1 \leqslant 2$$

$$0 \leqslant x_2 \leqslant 2.$$

Soluţia acestei probleme este $\mathbf{x}^1 = (2, 1)$. Unind punctul $\mathbf{x}^1$ cu P_0 obţinem punctul de intersecţie $P_2 = (1, 1)$.

Tangenta la parabola $x_1^2 - x_2 = 0$ este:

$$2x_1 - x_2 - 1 = 0.$$

Următoarea problemă de programare liniară este deci:

$$\max\ (x_1 + x_2)$$

în condiţiile

$$3x_1 + 4x_2 \leqslant 10$$

$$2x_1 - x_2 \leqslant 1$$

$$0 \leqslant x_1 \leqslant 2$$

$$0 \leqslant x_2 \leqslant 2$$

Soluţia ei este $\mathbf{x}^2 = \left(\frac{14}{11}, \frac{17}{11}\right)$.

Din Figura 12.2., se vede că soluţia exactă a problemei iniţiale este punctul de intersecţie al cercului cu parabola:

$$\mathbf{x}^* = \left(\sqrt{\frac{\sqrt{17}-1}{2}};\ \frac{\sqrt{17}-1}{2}\right) \approx (1{,}2;\ 1{,}5)$$

Prin urmare iteraţia $\mathbf{x}^2$ aproximează destul de bine soluţia exactă.

Aproape simultan cu Kleibohm, tot în anul 1966, ZOUTENDIJK G. [81], a publicat metoda sa, denumită *metoda MFD* (de la „Modified feasible

Scanned with OKEN Scanner

direction"), care seamănă oarecum cu metoda lui Kleibohm. Însă pe cînd în metoda lui Kleibohm punctul admisibil P_0 se lasă fix, iar soluţiile aproximative $\mathbf{x}^0$, $\mathbf{x}^1$, ..., nu sînt soluţii admisibile, conducînd doar la limită la o soluţie admisibilă, care este totodată şi soluţia optimală a problemei, la Zoutendijk, se construieşte un şir de soluţii admisibile, convergent către soluţia optimă. Prin urmare în primul caz convergenţa are loc din exterior, iar în al doilea — din interior.

Unul din dezavantajele metodei lui Kleibohm este acela că *„migrarea" făcîndu-se mereu în direcţia punctului fix* P_0, convergenţa este destul de înceată.

Scanned with OKEN Scanner

CAPITOLUL III

CAZURI SPECIALE DE PROGRAMARE NELINIARĂ

§ 13. Minimizarea sumei modulelor funcţiilor liniare

Vom da acum cîteva cazuri speciale de programare neliniară la care conduc o serie de probleme practice. Vom începe cu problema minimizării unei funcţii ce reprezintă suma modulelor unor funcţii liniare. Astfel, să considerăm funcţiile

$$f_i(\mathbf{x}) = f_i(x_1, x_2, \ldots, x_n) = \mathbf{c}_i\mathbf{x}, \qquad (i = 1, 2, \ldots, p) \tag{13.1}$$

şi restricţiile

$$g_j(\mathbf{x}) \leqslant 0. \tag{13.2}$$

unde g_j sînt convexe.

Problema care se pune este următoarea: *să se minimizeze* funcţia

$$f(\mathbf{x}) = \sum_{i=1}^{p} |f_i(\mathbf{x})| \tag{13.3}$$

în condiţiile (13.2). Evident că problema de programare (13.3) — (13.2) este o problemă de programare convexă, întrucît funcţia de scop $f(\mathbf{x})$ este o funcţie convexă, *segmentar liniară*.

Prin introducerea unor variabile auxiliare, această problemă se poate reduce la o problemă de programare liniară dacă g_j sînt funcţii liniare. Astfel, punînd

$$|f_i(\mathbf{x})| \leqslant y_i, \qquad (i = 1, 2, \ldots, p)$$

atunci problema (13.3) — (13.2) se poate formula sub următoarea formă echivalentă: *să se minimizeze funcţia*

$$F(\mathbf{y}) = \sum_{i=1}^{p} y_i \tag{13.4}$$

Scanned with OKEN Scanner

în condițiile

$$
\begin{aligned}
& f_i(\mathbf{x}) - y_i \leqslant 0 \\
& -f_i(\mathbf{x}) - y_i \leqslant 0, \qquad (i = 1, 2, \ldots, p) \\
& g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots, m)
\end{aligned} \tag{13.5}
$$

care este evident o problemă de programare liniară. Se verifică ușor că această problemă este echivalentă cu problema de programare convexă (13.3)—(13.2).

§ 14. Aproximarea uniformă în domeniul complex

Să considerăm o funcție $f(t)$ definită și continuă pe o mulțime Q compactă din planul complex, pe care vrem să o aproximăm cu polinoame generalizate de forma

$$z_1 f_1(t) + z_2 f_2(t) + \ldots + z_n f_n(t), \quad (z_k = x_k + iy_k;\ k = 1, 2, \ldots, n),$$

unde $f_1, f_2, \ldots, f_n$ sînt funcții liniar independente date, continue pe Q, iar coeficienții z_k, $k = 1, 2, \ldots, n$, verifică condițiile

$$g_j(z_1, z_2, \ldots, z_n) \leqslant b_j, \qquad (j = 1, 2, \ldots, m), \tag{14.1}$$

unde $g_j(z_1, \ldots, z_n)$ sînt funcții de variabile complexe cu valori reale diferențiabile și convexe. Atunci aceste inegalități definesc un domeniu convex Ω din spațiul euclidian $2n$—dimensional al variabilelor $x_1, x_2, \ldots, x_n$; $y_1, y_2, \ldots, y_n$.

Restricțiile (14.1) pot fi de exemplu de forma

$$|\mathbf{Az}| \leqslant \mathbf{b} \tag{14.2}$$

unde **A** este o matrice de numere reale de tipul (m, n), iar **b** un vector de asemenea real.

Prin urmare, problema care se pune este următoarea: să se găsească un punct $\mathbf{z}^* = (z_1^*, z_2^*, \ldots, z_n^*)$ care verifică inegalitățile (14.1), numit *punctul lui Cebîșev*, pentru care are loc relația

$$\max_{t \in Q} \left| \sum_{k=1}^{n} z_k^* f_k(t) - f(t) \right| = \inf_{z \in \Omega} \max_{t \in Q} \left| \sum_{k=1}^{n} z_k f_k(t) - f(t) \right| \tag{14.3}$$

Scanned with OKEN Scanner

Deoarece funcţia

$$F(\mathbf{z}) = F(x_1, \ldots, x_n; y_1, \ldots, y_n) = \max_{t \in \Omega} \left| \sum_{k=1}^{n} z_k f_k(t) - f(t) \right|$$

este o funcţie convexă, după cum se poate verifica uşor, rezultă că această problemă este o problemă de programare convexă.

Deoarece funcţiile $f, f_1, \ldots, f_n$ sînt continue pe compactul Q, vom obţine o soluţie aproximativă a problemei considerate dacă o rezolvăm pe o reţea finită, destul de fină, $\{t_1, t_2, \ldots, t_p\}$ a compactului Q.

Dacă notăm

$$f_k(t_h) = \alpha_{hk} = a_{hk} + ib_{hk}, \; f(t_h) = \alpha_h$$

$$(h = 1, 2, \ldots, p; \quad k = 1, 2, \ldots, n),$$

vom ajunge la următoarea problemă de cea mai bună aproximaţie în domeniul complex.

Se dă sistemul de p ecuaţii liniare complexe

$$\Delta_h(\mathbf{z}) = \alpha_{h1} z_1 + \ldots + \alpha_{hn} z_n + \alpha_h = 0, \quad (h = 1, 2, \ldots, p) \tag{14.4}$$

şi sistemul de inegalităţi (14.1). Se cere să se găsească punctul cebîşevian al sistemului (14.4) supus restricţiilor (14.1), adică se cere să se găsească un punct $\mathbf{z}^* \in \Omega$, pentru care

$$\max_{(h)} |\Delta_h(z^*)| = \min_{z \in \Omega} \max_{(h)} |\Delta_h(\mathbf{z})|. \tag{14.5}$$

Întrucît funcţia $\max_{(h)} |\Delta_h(\mathbf{z})|$ este o funcţie convexă, rezultă că problema (14.5)—(14.1) este o problemă de programare convexă şi deci poate fi rezolvată cu oricare din metodele prezentate mai sus.

§ 15. Punctul cel mai apropiat de un sistem de plane

Vom considera o altă problemă importantă, care conduce la programarea neliniară. Pentru aceasta să considerăm în spaţiul euclidian n-dimensional un sistem de hiperplane $(n-k)$-dimensionale

$$\alpha_i : \mathbf{A}^i \mathbf{x} + \mathbf{b}^i = \mathbf{0}, \qquad (i = 1, 2, \ldots, m), \tag{15.1}$$

unde $\mathbf{A}^i = (a^i_{jk})$ este o matrice de tipul $(n-k, n)$, iar $\mathbf{b}^i = (b^i_1, \ldots, b^i_{n-k})'$ este un vector coloană. Se presupune că hiperplanele din sistemul (15.1)

Scanned with OKEN Scanner

sînt două cîte două ortogonale şi ecuaţiile lor sînt aduse la forma normală. Atunci

$$\rho(\mathbf{x}^0,\ \alpha_i) = \sqrt{(\mathbf{a}_1^i\mathbf{x}^0 + b_1^i)^2 + \ldots + (\mathbf{a}_{n-k}^i\mathbf{x}^0 + b_{n-k}^i)^2}$$

reprezintă distanţa dintre punctul $\mathbf{x}^0$ şi hiperplanul α_i. Considerăm acum problema găsirii punctului, care se abate cel mai puţin de la sistemul de hiperplane(15.1), adică problema găsirii unui punct $\mathbf{x}^* \in \mathbf{R}^n$ pentru care

$$\max_{(i)} \rho(\mathbf{x}^*,\ \alpha_i) = \inf_{\mathbf{x}\in\mathbf{R}^n} \max_{(i)} \rho(\mathbf{x},\ \alpha_i). \tag{15.2}$$

Se poate verifica uşor că funcţia

$$f(\mathbf{x}) = \max_{(i)} \rho(\mathbf{x},\ \alpha_i)$$

este o funcţie convexă, aşa încît problema (15.2)—(15.1) este o problemă de programare convexă cu restricţii liniare.

Pentru $k = n - 1$ hiperplanele α_i sînt hiperplane obişnuite şi sistemele (15.1) se reduc la cîte o singură ecuaţie. În acest caz problema se reduce la găsirea punctului din spaţiul euclidian n-dimensional care se abate cel mai puţin de la un sistem dat de hiperplane, adică conduce la problema clasică a găsirii *punctului lui Cebîşev* pentru un sistem incompatibil de ecuaţii liniare.

În cazul cînd $k = 0$, atunci hiperplanele α_i se reduc la punctele din spaţiul $\mathbf{R}^n$ şi astfel, problema (15.2) — (15.1) se reduce la găsirea unui punct din spaţiul $\mathbf{R}^n$ care se abate cel mai puţin de la un sistem dat de puncte din $\mathbf{R}^n$, problemă care a fost considerată pentru prima dată de

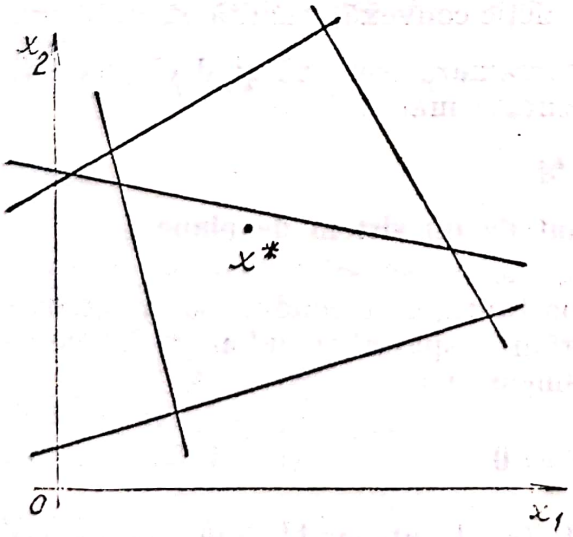

Figura 15.1

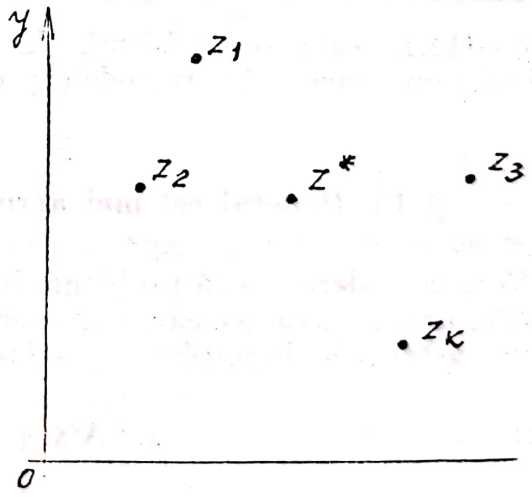

Figura 15.2.

ZUHOVIȚKI S. I. (vezi [86]) în anul 1951, care a dat și un algoritm pentru determinarea acestui punct.

În cazul plan, această problemă este strîns legată de problema determinării polinomului lui Cebîșev de gradul întîi care se abate cel mai puțin de la zero pe o mulțime finită din plan. În Figura 15.1 este ilustrat cazul plan al punctului lui Cebîșev al unui sistem de drepte din plan, iar în Figura 15.2 este dat punctul lui Cebîșev al unui sistem de puncte din plan, adică punctul care se abate cel mai puțin de la un sistem dat de puncte din plan. Ultima problemă are tangență cu o problemă de tehnică legată de reprezentarea unei mase date pe un sistem de puncte de sprijin, astfel încît *deschiderea sistemului să fie minimă*, adică distanța de la centrul de greutate al sistemului la cel mai depărtat punct de sprijin să fie minimă. Această problemă a fost rezolvată în 1962 de MARUȘCIAC, I., MUNTEANU E. și RUS T. [48]. În lucrare s-a dat și schema logică a unui program pentru o mașină electronică de calcul.

§ 16. Programarea segmentar-liniară

Vom insista ceva mai mult asupra unei probleme speciale de programare convexă, numită problemă de programare segmentar-liniară, deoarece în acest caz se pot da algoritmi mult mai eficienți din punct de vedere practic, decît în cazul general.

Prin programare segmentar-liniară se înțelege în general o problemă de programare convexă în care funcția de scop este liniară pe segmente. Această problemă este o problemă intermediară între problema de programare liniară și problema de programare convexă. Există o serie întreagă de probleme practice care conduc la problema de programare segmentar-liniară (vezi [22]).

Fie date funcțiile liniare

$$L_j(\mathbf{x}) = L_j(x_1,\ x_2,\ \ldots,\ x_n) = \sum_{i=1}^{n} a_{ji}x_i - b_j\,, \qquad (j = 1,\ 2,\ \ldots,\ s).$$

Se numește *funcție segmentar-liniară* de variabilele $x_1,\ x_2,\ \ldots,\ x_n$, o funcție definită astfel:

$$F(\mathbf{x}) = \max_{(j)}\ [L_j(\mathbf{x})]. \tag{16.1}$$

Se vede imediat că funcția $F(\mathbf{x})$ definită de (16.1) este o funcție convexă.

Dacă acum considerăm mai multe funcții g_j segmentar-liniare, atunci inegalitățile

$$g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1,\ 2,\ \ldots,\ m) \tag{16.2}$$

determină un domeniu poliedral Ω.

Scanned with OKEN Scanner

Prin *problema generală de programare segmentar-liniară* se înţelege următoarea problemă:

Să se minimizeze funcţia

$$F(\mathbf{x})$$

în condiţiile

$$g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots, m),$$

unde F şi g_j sînt funcţii segmentar-liniare.

Dacă considerăm vector-funcţia $\mathbf{G}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x}))$, atunci problema se enunţă în felul următor: *să se minimizeze funcţia*

$$F(\mathbf{x})$$

cînd

$$\mathbf{G}(\mathbf{x}) \leqslant \mathbf{0}.$$

Într-o serie de cazuri este natural să se separe din condiţiile (16.2) cele liniare date sub forma unor egalităţi, şi atunci problema de programare segmentar-liniară se enunţă sub următoarea formă: *să se minimizeze funcţia*

$$F(\mathbf{x})$$

în condiţiile

$$\text{(16.3)} \qquad \begin{aligned} &\mathbf{G}(\mathbf{x}) \leqslant \mathbf{0} \\ &\mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0} \quad \text{şi} \quad \mathbf{x} \in \Omega, \end{aligned}$$

unde Ω este o mulţime poliedrală oarecare din spaţiul euclidian $\mathbf{R}^n$.

Dăm acum cîteva cazuri speciale care intervin în practică.

Problema I. Să se minimizeze funcţia

$$F(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i)$$

în condiţiile

$$\mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0}$$

$$\alpha \leqslant \mathbf{x} \leqslant \beta$$

Scanned with OKEN Scanner

unde $\mathbf{A}$ *este o matrice dată de tipul* (m, n), $\mathbf{b}$ *o matrice coloană de tipul* $(m, 1)$, *iar* α *şi* β *sînt matrici coloană de tipul* $(n, 1)$. Funcţiile $_i(x_i)$ fiind funcţii de o variabilă segmentar-liniare.

Problema II. Să se minimizeze funcţia

$$F(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i)$$

în condiţiile

$$\sum_{i=1} g_{ji}(x_i) \leqslant 0, \qquad (j = 1, 2, \ldots, m)$$

$$\alpha \leqslant \mathbf{x} \leqslant \beta,$$

unde f_i *şi* g_{ji} *sînt funcţii segmentar-liniare de variabila* x_i.

Problema III. Să se minimizeze funcţia

$$F(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i)$$

în condiţiile

$$\sum_{i=1}^{n} g_{ji}(x_i) \leqslant 0, \qquad (j = 1, 2, \ldots, m_1)$$

$$\sum_{i=1}^{n} a_{ji}x_i - b_i = 0, \qquad (j = m_1 + 1, \ldots, m)$$

$$\alpha \leqslant \mathbf{x} \leqslant \beta.$$

Se vede că problema III include atît problema II cît şi problema I.

Ideea pe care se bazează metoda de rezolvare a acestor probleme de programare segmentar-liniare este aceea de a reduce numărul mare de variabile şi restricţii, „complicînd puţin" funcţia de scop. Acest lucru se poate realiza atunci cînd printre restricţii figurează şi unele restricţii simple, ca de exemplu restricţii de mărginire a variabilelor independente.

Se pot pune în evidenţă o serie de probleme tehnice, economice sau de biologie, care se formulează ca probleme de programare segmentar-liniare. Astfel de probleme sînt, de exemplu, planificarea calendaristică a producţiei, problema reglării automate, probleme legate de construirea schemelor în teoria siguranţei etc.

În unele cazuri este foarte util să se reducă la problema de programare segmentar-liniară o problemă de programare convexă. Pentru aceasta este suficient să se aproximeze funcţiile convexe cu funcţii liniare pe segmente.

Scanned with OKEN Scanner

De asemenea orice problemă de programare convexă în numere întregi se poate reduce la o problemă de programare segmentar-liniară echivalentă (vezi GOLSTEIN. E. G. și IUDIN. D. B. [22]).

Pentru a vedea cum se poate reduce o problemă mare de programare liniară la o problemă echivalentă mult mai mică de programare segmentar-liniară, considerăm următorul exemplu de problemă de programare, care intervine în repartizarea optimă a producției pe ramuri de producție.

Astfel, să considerăm următoarea problemă: *să se minimizeze funcția*

$$L(\mathbf{x}) = \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n} \sum_{j=1}^{l_i} s_{ij} x_{ij}, \qquad s_{ij} \geqslant 0 \tag{16.4}$$

în condițiile

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{16.5}$$

$$\alpha \leqslant \mathbf{x} \leqslant \beta, \tag{16.6}$$

$$\left.\begin{aligned} x_i &= d_{ij} + x_{ij} \\ x_{ij} &\geqslant 0 \end{aligned}\right\}, \quad (i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m). \tag{16.7}$$

Această problemă se caracterizează prin faptul că restricțiile sînt grupate în patru categorii. În primele două restricții intervin numai variabilele x_i, iar marea majoritate a variabilelor x_{ij} intervin numai în (16.7). De aceea, ar fi nerațional să se folosească metoda obișnuită de rezolvare a problemei de programare liniară. Se va putea însă reduce această problemă la o problemă de programare segmentar-liniară mult mai simplă.

Astfel, din (16.6) rezultă că numerele d_{ij} sînt cuprinse în intervalul $[\alpha_i, \beta_i]$, deci putem presupune că ele au fost astfel notate încît să avem

$$\alpha_i < d_{i1} < d_{i2} < \ldots < d_{il_i} < \beta_i.$$

De aceea prin complicarea funcției de scop, vom elimina condițiile (16.7) și variabilele x_{ij}, considerînd funcțiile

$$f_i(x_i) = \begin{cases} c_i x_i & \alpha_i \leqslant x_i \leqslant d_{i1} \\ (c_i + s_{i1})(x_i - d_{i1}) + f_i(d_{i1}) & d_{i1} \leqslant x_i \leqslant d_{i2} \\ \ldots\ldots\ldots\ldots\ldots\ldots & \ldots\ldots \\ \left(c_i + \sum_{j=1}^{l_i} s_{ij}\right)(x_i - d_{il_i}) + f_i(d_{il_i}), & d_{il_i} \leqslant x_i \leqslant \beta_i. \end{cases}$$

Atunci se poate verifica ușor că problema de programare liniară (16.4) — (16.7) este echivalentă cu următoarea problemă de programare segmentar liniară: *să se minimizeze funcția*

$$F(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i) \tag{16.8}$$

în condițiile (16.5) — (16.6).

Scanned with OKEN Scanner

Întrucît $s_{ik} \geqslant 0$, toate funcţiile $f_i(x_i)$ sînt funcţii convexe şi prin urmare, problema de programare (16.8) este problemă de programare convexă, segmentar-liniară.

Criteriu de optim pentru problema de programare segmentar-liniară.

Să considerăm problema de programare segmentar-liniară enunţată sub forma: *să se minimizeze funcţia*

$$F(\mathbf{x}) \tag{16.9}$$

în condiţiile

$$\mathbf{G}(\mathbf{x}) \leqslant \mathbf{0}; \tag{16.10}$$

$$\mathbf{A}\mathbf{x} = \mathbf{b}; \tag{16.11}$$

$$\mathbf{x} \in \Omega, \tag{16.12}$$

unde F *şi* g_i $(\mathbf{G}(\mathbf{x})' = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x}))$ *sînt funcţii segmentar-liniare,* $\mathbf{A}$ *este o matrice de tipul* $(m - m_1, n)$; $\mathbf{b}$*-un vector coloană,* $m - m_1$*-dimensional, iar* Ω *este un poliedru din* $\mathbf{R}^n$.

Pentru această problemă se poate da un criteriu de optim relativ simplu, care permite construirea unui algoritm pentru găsirea soluţiei problemei într-un număr finit de paşi.

TEOREMA 16.1. *Un punct* $\mathbf{x}^*$ *este un punct optimal pentru problema* (16.9) — (16.12) *dacă şi numai dacă există un vector m-dimensional*

$$\Lambda = (\Lambda_1, \Lambda_2), \quad \Lambda_1 = (\lambda_1, \ldots, \lambda_{m_1}) \geqslant 0, \quad \Lambda_2 = (\lambda_{m_1+1}, \ldots, \lambda_m)$$

astfel încît funcţia

$$\varphi(\mathbf{x}, \Lambda) = F(\mathbf{x}) + \Lambda_1\mathbf{G}(\mathbf{x}) + \Lambda_2(\mathbf{A}\mathbf{x} - \mathbf{b}) \tag{a}$$

îşi atinge minimul în punctul $\mathbf{x}^*$ *şi*

$$\Lambda_1\mathbf{G}(\mathbf{x}^*) = \mathbf{0}. \tag{b}$$

Vom demonstra mai întîi *suficienţa*. Astfel să presupunem că există vectorii Λ_1 şi Λ_2 verificînd condiţiile a) şi b). Atunci din condiţia a) rezultă

$$\varphi(\mathbf{x}^*, \Lambda) \leqslant \varphi(\mathbf{x}, \Lambda), \qquad \mathbf{x} \in \Omega$$

sau, ceea ce este acelaşi lucru, avem

$$F(\mathbf{x}^*) + \Lambda_1\mathbf{G}(\mathbf{x}^*) + \Lambda_2(\mathbf{A}\mathbf{x}^* - \mathbf{b}) \leqslant F(\mathbf{x}) + \Lambda_1\mathbf{G}(\mathbf{x}) + \Lambda_2(\mathbf{A}\mathbf{x} - \mathbf{b}), \qquad \mathbf{x} \in \Omega.$$

Scanned with OKEN Scanner

Evident că $\mathbf{Ax^* - b = 0}$.
Din condiţia b)

$$\Lambda_1 \mathbf{G}(\mathbf{x^*}) = \mathbf{0}.$$

Însă oricare ar fi $\mathbf{x}$ o soluţie admisibilă a problemei de programare (16.9) — (16.12), rezultă că avem

$$\mathbf{Ax - b = 0} \text{ şi } \qquad \Lambda_1 \mathbf{G}(\mathbf{x}) \leqslant \mathbf{0},$$

de aceea rezultă că

$$F(\mathbf{x^*}) \leqslant F(\mathbf{x}),$$

adică $\mathbf{x^*}$ este un punct optimal al problemei noastre.

Necesitatea se demonstrează ceva mai greu, de aceea nu o vom da aici (ea poate fi găsită în [22]).

Aplicăm acum teorema de mai sus problemei tipice de programare segmentar-liniară enunţată mai sus. Să considerăm, de exemplu, *Problema I*: să se minimizeze funcţia:

$$F(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i) \tag{16.13}$$

în condiţiile

$$\mathbf{Ax - b = 0} \tag{16.14}$$

$$\alpha \leqslant \mathbf{x} \leqslant \beta. \tag{16.15}$$

Din criteriul de mai sus rezultă că punctul $\mathbf{x^*}$ este un punct optimal pentru problema (16.13) — (16.15) atunci şi numai atunci cînd există un vector Λ pentru care

$$\varphi(\mathbf{x}, \Lambda) = F(\mathbf{x}) + \Lambda(\mathbf{Ax - b})$$

îşi atinge minimul în $\mathbf{x^*}$ pe mulţimea definită de inegalităţile

$$\alpha_i \leqslant x_i \leqslant \beta_i.$$

Prin urmare punctul $\mathbf{x^*}$ este optimal dacă şi numai dacă există un vector Λ astfel încît

$$F(\mathbf{x^*}) + \Lambda(\mathbf{Ax^* - b}) \leqslant F(\mathbf{x}) + \Lambda(\mathbf{Ax - b})$$
$$\alpha \leqslant \mathbf{x} \leqslant \beta$$

Scanned with OKEN Scanner

care se transcrie sub forma

$$\sum_{i=1}^{n} f_i(x_i^*) + \sum_{i=1}^{n} (\Lambda a_i)\, x_i^* \leqslant \sum_{i=1}^{n} f_i(x_i) + \sum_{i=1}^{n} (\Lambda a_i) x_i, \tag{16.16}$$

$$\alpha_i \leqslant x_i \leqslant \beta_i.$$

Fie $x_k^* < \beta_k$. Punem atunci $x_k = x_k^* + \varepsilon$ $(\varepsilon > 0)$, iar restul variabilelor le scriem neschimbate, $x_i = x_i^*$, $i \neq k$.

În acest caz inegalitatea (16.16) devine

$$-(\Lambda a_k)\varepsilon \leqslant f_k(x_k^* + \varepsilon) - f_k(x_k^*),$$

de unde obţinem

$$f_k^{(+)}(x_k^*) \geqslant -(\Lambda a_k), \tag{16.17}$$

unde $f^{(+)}(x_k^*)$ reprezintă derivata la dreapta în punctul x_k^*.

Dacă acum $x_k^* > \alpha_k$, atunci punem $x_k = x_k^* - \varepsilon$ $(\varepsilon > 0)$, iar restul variabilelor din nou neschimbate, $x_i = x_i^*$, $i \neq k$. Atunci din (16.16) vom obţine în mod analog inegalitatea

$$f_k^{(-)}(x_k^*) \leqslant -(\Lambda a_k). \tag{16.18}$$

Întrucît Λ poate avea orice semn, în (16.17), (16.18) se poate omite semnul, şi astfel pentru x_i^* cuprins între α_i şi β_i rezultă inegalitatea

$$f_i^{(-)}(x_i^*) \leqslant (\Lambda a_i) \leqslant f_i^{(+)}(x_i^*). \tag{16.19}$$

Dacă $x_i^* = \alpha_i$, atunci avem

$$f_i^{(+)}(x_i^*) \geqslant (\Lambda a_i), \tag{16.20}$$

iar dacă $x_i^* = \beta_i$, avem

$$f_i^{(-)}(x_i^*) \leqslant (\Lambda a_i). \tag{16.21}$$

În concluzie, un punct $\mathbf{x}^*$ *este optimal pentru problema I, dacă şi numai dacă există un vector* Λ *astfel încît să fie verificate condiţiile* (16.19)—(16.21).

Se pot transcrie criterii analoage şi pentru celelalte două probleme (vezi GOLSTEIN, E. G. şi IUDIN, D. B. [22], pag. 454).

Scanned with OKEN Scanner

Folosind criteriul obţinut pentru problema I se poate construi un algoritm, asemănător cu metoda simplex. În prima etapă a algoritmului se verifică dacă punctul obţinut la iteraţia precedentă este sau nu optimal. Dacă punctul este optimal atunci procesul se termină, iar în caz contrar se trece la etapa a doua a iteraţiei, prin care fie că se construieşte un nou punct $\mathbf{x}^{k+1}$ pentru care valoarea funcţiei de scop este mai mică decît în punctul $\mathbf{x}^k$, fie se stabileşte că funcţia de scop $F(\mathbf{x})$ este nemărginită pe domeniul restricţiilor.

Se demonstrează [22] că totdeauna algoritmul conduce la soluţia problemei I după un număr finit de paşi. Se poate da chiar o delimitare superioară pentru numărul maxim de iteraţii necesare pentru a se ajunge la soluţia problemei.

Celelalte două probleme se pot trata la fel, iar algoritmii lor nu diferă prea mult de cel care corespunde problemei I.

§ 17. Programarea monotonă

O funcţie f definită pe o mulţime convexă $\Omega \in \mathbf{R}^n$ se numeşte *monotonă pe* Ω, *dacă oricare ar fi punctele* $\mathbf{x}'$, $\mathbf{x}'' \in \Omega$ *şi oricare ar fi* $t \in [0, 1]$, are loc relaţia

$$\min [f(\mathbf{x}'), f(\mathbf{x}'')] \leqslant f(t\mathbf{x}' + (1-t)\mathbf{x}'') \leqslant \max [f(\mathbf{x}'), f(\mathbf{x}'')]. \tag{17.1}$$

În cazul unei funcţii f de o variabilă reală definită pe un interval, această definiţie coincide cu definiţia funcţiei monotone, ceea ce justifică denumirea de funcţie monotonă.

Mai observăm că din definiţia funcţiilor cvasi concave şi cvasi convexe pe un domeniu convex Ω rezultă că o *funcţie* f *este monotonă pe* Ω *dacă este în acelaşi timp şi cvasi convexă şi cvasi concavă pe* Ω.

Definiţia 17.1. Se numeşte programare monotonă o programare de forma

$$\min(\max)\{f(\mathbf{x}) | \mathbf{x} \in \Omega\},$$

unde Ω *este un domeniu convex din* $\mathbf{R}^n$, *iar funcţia de scop* f, *este monotonă pe* Ω.

Vom prezenta în acest paragraf cîteva dintre rezultatele mai importante relative la programarea monotonă.

Vom considera problema de programare monotonă cu restricţii liniare, de forma

$$\max \{f(\mathbf{x}) | \mathbf{A}\mathbf{x} \leqslant \mathbf{b}, \ \mathbf{x} \geqslant \mathbf{0}\}. \tag{17.2}$$

Scanned with OKEN Scanner

Se ştie [34] că domeniul convex închis

$$\Omega = \{\mathbf{x} \in \mathbf{R} \mid \mathbf{Ax} \leqslant \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}\} \neq \emptyset,$$

este generat de puncte de forma

$$\mathbf{x} = \sum_{s=1}^{p} \lambda_s P_s + \sum_{t=1}^{r} \mu_t Q_t, \tag{17.3}$$

unde

$$\lambda_s,\ \mu_t \geqslant 0,\quad p > 0,\quad r \geqslant 0,\quad \Sigma \lambda_s = 1,$$

P_s sînt vîrfurile poliedrului Ω care sînt soluţii ale sistemului

$$\mathbf{Ax} \leqslant \mathbf{b},\qquad \mathbf{x} \geqslant \mathbf{0}, \tag{17.4}$$

iar Q_t soluţii ale sistemului

$$\mathbf{Ax} \leqslant 0,\quad \mathbf{x} \geqslant \mathbf{0}.$$

Menţionăm că aici Ω poate fi şi nemărginit.

Se poate verifica uşor că oricare ar fi $\mathbf{x} \in \Omega$, din (17.3) rezultă

$$\min_{s,t} \left\{ f(P_s),\ \lim_{\mu_t \to \infty} f(\mu_t Q_t) \right\} \leqslant f(\mathbf{x}) \leqslant \tag{17.5}$$

$$\leqslant \max_{s,t} \left\{ f(P_s),\ \lim_{\mu_t \to \infty} f(\mu_t Q_t) \right\}.$$

Într-adevăr, să adăugăm la inegalităţile (17.4) inegalitatea

$$x_1 + x_2 + \ldots + x_n \leqslant B, \tag{17.6}$$

unde B este o constantă suficient de mare astfel ca punctele P_s să se afle de aceeaşi parte a hiperplanului

$$x_1 + x_2 + \ldots + x_n = B,$$

cu originea axelor de coordonate. Mulţimea soluţiilor sistemului de inegalităţi (17.4), (17.6) este un poliedru mărginit Ω' cu vîrfurile P_s, $s = 1, 2, \ldots, p$, şi $R_t(B)$, $t = 1, 2, \ldots, r$. Un punct $\mathbf{x} \in \Omega$ se poate scrie sub forma

$$\mathbf{x} = \sum_{s=1}^{p} \gamma_s P_s + \sum_{t=1}^{r} \delta_t R_t(B), \tag{17.7}$$

Scanned with OKEN Scanner

unde

$$\gamma_s,\ \delta_t \geqslant 0;\quad \Sigma\gamma_s + \Sigma\delta_t = 1.$$

Din definiţia funcţiei monotone avem

$$\min_{s,t}\{f(P_s),\ f[R_t(B)]\} \leqslant f(\mathbf{x}) \leqslant \max_{s,t}\{f(P_s),\ f[R_t(B)]\}.$$

De aici prin trecere la limită pentru $B \to +\infty$, se obţine (17.5).

Din (17.5) se vede că mulţimea soluţiilor optimale (maxim, sau minim) Ω^* a problemei de programare monotonă, dacă este nevidă, conţine cel puţin un punct P_s, sau punctul de la infinit sau şi una şi alta şi astfel este o mulţime poliedrală convexă închisă (mărginită sau nu) şi este formată din puncte $\mathbf{x}^*$ de forma

$$\mathbf{x}^* = \sum_{s=1}^{p_1} \lambda_s P_s + \sum_{t=1}^{r_1} \mu_t Q_t \tag{17.8}$$

unde

$$\lambda_s,\ \mu_t \geqslant 0, \qquad \Sigma\lambda_s = 1.$$

Aici punctele P_s şi Q_t s-au presupus a fi punctele în care are loc egalitatea în (17.5).

Se ştie [80] că *funcţia raţională*

$$f(\mathbf{x}) = \frac{c_1x_1 + c_2x_2 + \ldots + c_nx_n + \alpha}{d_1x_1 + d_2x_2 + \ldots + d_nd_n + \beta} = \frac{\mathbf{cx} + \alpha}{\mathbf{dx} + \beta}$$

este monotonă pe orice domeniu convex $D \subset \mathbf{R}^n$, *dacă*

$$\mathbf{dx} + \beta = d_1x_1 + d_2x_2 + \ldots + d_nx_n + \beta > 0, \qquad \mathbf{x} \in \mathbf{R}^n.$$

Vom demonstra acest rezultat în cazul cînd şi $\mathbf{cx} + \alpha > 0$, $\mathbf{x} \in D$. Fie deci

$$\mathbf{x}',\ \mathbf{x}'' \in D,\ \mathbf{x}' \neq \mathbf{x}''\ \text{şi}\ \mathbf{dx} + \beta > 0,\ \mathbf{x} \in D,$$

şi $t \in [0, 1]$. Presupunem că $f(\mathbf{x}') > f(\mathbf{x}'')$. Atunci, deoarece

$$(\mathbf{dx}' + \beta)(\mathbf{dx}'' + \beta) > 0,$$

avem

$$(\mathbf{cx}' + \alpha)(\mathbf{dx}'' + \beta) - (\mathbf{cx}'' + \alpha)(\mathbf{dx}' + \beta) \geqslant 0.$$

Scanned with OKEN Scanner

Înmulţind această inegalitate cu $(1 - t)$ şi adunînd şi scăzînd

$$t(\mathbf{cx}' + \alpha)(\mathbf{dx}' + \beta)$$

obţinem

$$(\mathbf{cx}' + \alpha)[t(\mathbf{dx}' + \beta) + (1 - t)(\mathbf{dx}'' + \beta)] -$$
$$- (\mathbf{dx}' + \beta)[t(\mathbf{cx}' + \alpha) + (1 - t)(\mathbf{cx}'' + \alpha)] \geqslant 0,$$

adică

$$(\mathbf{cx}' + \alpha)(\mathbf{dx}^0 + \beta) - (\mathbf{dx}' + \beta)(\mathbf{cx}^0 + \alpha) \geqslant 0,$$

unde

$$\mathbf{x}^0 = t\mathbf{x}' + (1 - t)\mathbf{x}''.$$

Întrucît $\mathbf{dx} + \beta > 0$, rezultă

$$f(\mathbf{x}^0) \leqslant f(\mathbf{x}').$$

În mod analog se demonstrează şi inegalitatea $f(\mathbf{x}^0) \geqslant f(\mathbf{x}'')$.

O programare monotonă în care restricţiile sînt liniare iar funcţia de scop este liniar-raţională se numeşte *programare liniar-raţională.*

Pentru ilustrare să considerăm următorul exemplu :

Exemplul 17.1.

$$\max f(\mathbf{x}) = \max \frac{13x_1 - 6x_2 + 66}{x_1 + 6x_2 + 18}$$

în condiţiile

$$\begin{aligned} x_1 - 6x_2 &\leqslant 6 \\ -x_1 + 6x_2 &\leqslant 10 \\ -3x_1 + 2x_2 &\leqslant 6, \quad x_1 \geqslant 0, \quad x_2 \geqslant 0. \end{aligned}$$

Se vede direct că pentru $x_1 \geqslant 0$, $x_2 \geqslant 0$ numitorul fracţiei $f(\mathbf{x})$ este pozitiv şi astfel funcţia de scop f este monotonă pe domeniul soluţiilor admisibile.

Domeniul soluţiilor admisibile este nemărginit (vezi Figura 17.1). De aceea vom adăuga restricţia

$$x_1 + x_2 \leqslant B, \qquad (B > 8).$$

Scanned with OKEN Scanner

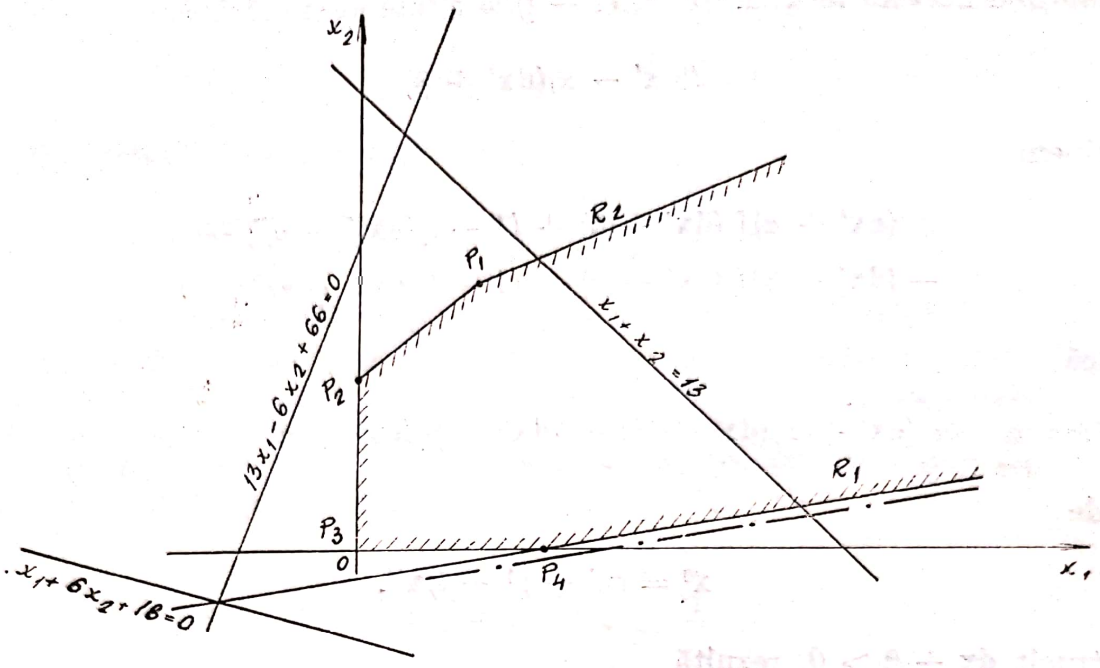

Figura 17.1

Se constată că vîrfurile poligonului sînt:

$$P_1(2,\ 6),\quad P_2(0,\ 3),\quad P_3(0,\ 0),\quad P_4(6,\ 0)$$

Punctele de intersecţie ale dreptei $x_1 + x_2 = B$ cu frontiera domeniului soluţiilor admisibile sînt:

$$R_1\left(\frac{6B+6}{7}, \frac{B-6}{7}\right),\quad R_2\left(\frac{2B-10}{3}, \frac{B+10}{3}\right).$$

Avem $f(P_1) = 1$, $f(P_2) = \frac{4}{3}$, $f(P_3) = \frac{11}{3}$, $f(P_4) = 6$,

$$f(R_1) = 6,\ \lim_{B\to\infty} f(R_1) = 6,\ f(R_2) = \frac{5}{2} - \frac{63}{2B+26},\ \lim_{B\to\infty} f(R_2) = \frac{5}{2}.$$

Prin urmare

$$\max f(\mathbf{x}) = f(P_4) = \lim_{B\to\infty} (R_1) = 6,$$

deci mulţimea soluţiilor optime este formată din punctele

$$\mathbf{x} = P_4 + \mu Q_1, \qquad \mu \geqslant 0,$$

unde $Q_1(6,\ 1)$ este vectorul director al dreptei P_4R_1.

Scanned with OKEN Scanner

Observăm că punctul R_1 se găseşte pe dreapta $x_1 - 6x_2 = 6$ care trece prin punctul comun al intersecţiei dreptelor

$$13x_1 - 6x_2 + 66 = 0 \quad \text{şi} \quad x_1 + 6x_2 + 18 = 0.$$

Dacă se modifică un singur termen liber al funcţiei de scop, considerînd funcţia

$$f(\mathbf{x}) = \frac{13\,x_1 - 6x_2 + 66}{x_1 + 6x_2 + 24}$$

atunci maximul funcţiei (în aceleaşi restricţii) este atins într-un singur punct la infinit, care corespunde direcţiei $\mathbf{v} = (6,\ 1)$ căci în toate celelalte puncte din domeniul soluţiilor

$$f(\mathbf{x}) < \lim_{B \to \infty} f[R_1(B)] = 6.$$

Un algoritm pentru determinarea unei soluţii a problemelor de programare monotonă.

Vom prezenta acum un algoritm pentru găsirea unei soluţii (vîrf) a problemei de programare monotonă, elaborat de KUCHER B. M. [39]. Se consideră problema:

$$\max \{f(\mathbf{x}) | \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}\},$$

unde f este o funcţie monotonă şi indefinit diferenţiabilă pe

$$\Omega = \{\mathbf{x} \in \mathbf{R}^n | \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}\}.$$

Algoritmul este o adaptare a metodei simplex a lui Dantzig. Dacă notăm cu

$$\mathbf{k}_j = (a_{1j},\ a_{2j},\ \ldots,\ a_{mj})', \qquad (j = 1,\ 2,\ \ldots,\ n),$$

atunci sistemul

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \qquad (i = 1,\ 2,\ \ldots,\ m) \tag{17.9}$$

Scanned with OKEN Scanner

$$x_j \geqslant 0, \qquad (j = 1, 2, \ldots, n)$$

(17.10)

se scrie

$$\mathbf{b} = \sum_{j=1}^{n} \mathbf{k}_j x_j, \qquad x_j \geqslant 0, \qquad (j = 1, 2, \ldots, n)$$

Fie $\mathbf{x}^1 = (x_1, \ldots, x_m, 0, \ldots, 0)$ o soluție de pornire. Atunci vectorii $\mathbf{k}_1, \mathbf{k}_2, \ldots, \mathbf{k}_m$ reprezintă o bază a spațiului euclidian $\mathbf{R}^m$ și avem

$$\mathbf{b} = \sum_{j=1}^{m} x_j \mathbf{k}_j \qquad \mathbf{k}_h = \sum_{j=1}^{m} \alpha_{jh}\, \mathbf{k}_j \qquad (h = 1, 2, \ldots, n).$$

Pentru $t > 0$ arbitrar și $h > m$, avem

$$\mathbf{b} = \sum_{j=1}^{m} x_j\, \mathbf{k}_j - t\mathbf{k}_h + t\mathbf{k}_h = \sum_{j=1}^{m} (x_j - t\alpha_{jh})\mathbf{k}_j + t\mathbf{k}_h.$$

Punctul $\mathbf{x} = (x_1 - t\alpha_{1h}, \ldots, x_m - t\alpha_{mh}, 0, \ldots, 0, t, 0, \ldots, 0)$, unde t este situat în componenta h, aparține lui Ω dacă

$$0 \leqslant t \leqslant \min_j \frac{x_j}{\alpha_{jh}}, \qquad \alpha_{jh} > 0$$

și dacă toți $\alpha_{jh} \leqslant 0$, $j = 1, 2, \ldots, m$, pentru orice $t \geqslant 0$ (cazul cînd Ω este nemărginit).

Se aleg acum vectorii $\mathbf{k}_h$, $h = m + 1, \ldots, n$, astfel încît

$$f(\mathbf{x}) - f(\mathbf{x}^1) > 0 \tag{17.11}$$

(dacă acest lucru e posibil).

Aplicînd formula lui Taylor funcției f în vecinătatea punctului $\mathbf{x}^1$, avem

$$f(\mathbf{x}) - f(\mathbf{x}^1) = t\left[c_h - \sum_{j=1}^{m} c_j\, \alpha_{jh}\right] + t^2 d^2 f(\xi), \tag{17.12}$$

unde

$$c_j = \frac{\partial f(\mathbf{x}^1)}{\partial x_j}, \qquad (j = 1, 2, \ldots, n)$$

$$\xi = \mathbf{x}^1 + \mu(\mathbf{x} - \mathbf{x}^1), \qquad 0 < \mu < 1.$$

Scanned with OKEN Scanner

Din (17.12) este clar că inegalitatea (17.11) are loc dacă

$$c_h - \sum_{j=1}^{m} c_j \alpha_{jh} > 0. \tag{17.13}$$

Din monotonia lui f rezultă că dacă (17.13) are loc pentru $h > m$, atunci

$$f(\mathbf{x}^2) > f(\mathbf{x}^1),$$

unde

$$\mathbf{x}^2 = (x_1 - \tau_h \alpha_{1h}, \ \ldots, \ x_m - \tau_h \alpha_{mh}, \ 0, \ \ldots, \ 0, \ \tau_h, \ 0 \ \ldots, \ 0),$$

$$\tau_h = \min_j \frac{x_j}{\alpha_{jh}}, \quad \alpha_{jh} > 0, \qquad (j = 1, \ 2, \ \ldots, \ m).$$

Dacă acum, odată cu (17.13) pentru un anumit $h > m$, $\alpha_{jh} > 0$ pentru orice $j = 1, \ 2, \ \ldots, \ m$, atunci $\lim_{\tau_h \to \infty} f(\mathbf{x}^2) > f(\mathbf{x}^1)$.

Prin urmare are loc următorul criteriu de optim

TEOREMA 17.1. *Dacă în* $\mathbf{x}^1$ $\nabla f(\mathbf{x}) \neq 0$, f *este indefinit diferențiabilă pe* Ω *și*

$$c_h - \sum_{j=1}^{m} c_j \alpha_{jh} \leqslant 0, \qquad (h = 1, \ 2, \ \ldots, \ n). \tag{17.14}$$

atunci $\mathbf{x}^1$ *este soluția optimă, adică* $\mathbf{x}^1 \in \Omega^*$.

Schema de calcul este aceeași în mare ca în cazul algoritmului simplex cu singura deosebire că aici la fiecare pas se verifică criteriul de optim (17.14).

Pentru ilustrare considerăm următorul exemplu:

Exemplul 17.2. Să se maximizeze funcția

$$f(x_1, \ x_2) = \frac{(1 - \sqrt{1 + x_1 x_2})^2}{x_1^2}$$

în condițiile:

$$\begin{aligned}
&-2x_1 + x_2 \leqslant 10\\
&-4x_1 + x_2 \leqslant 4\\
&-5x_1 + x_2 \leqslant 2\\
&-2x_1 - x_2 \leqslant -9\\
&-\ x_1 - x_2 \leqslant -7\\
&\frac{1}{4} x_1 - x_2 \leqslant -1, \quad x_1 \geqslant 0, \quad x_2 \geqslant 0.
\end{aligned}$$

Scanned with OKEN Scanner

Se poate verifica că funcţia din acest exemplu este monotonă şi deci se poate aplica algoritmul descris.

Introducînd variabilele auxiliare şi renumerotîndu-le, sistemul de inegalităţi de mai sus se scrie, după unele transformări simple, sub forma [39]:

$$\begin{aligned}
x_1 &- x_7 + x_8 = 2\\
x_2 &+ x_7 - 2x_8 = 5\\
x_3 &+ 5x_7 - 9x_8 = 14\\
x_4 &- 3x_7 + 4x_8 = 9\\
x_5 &- 5x_7 + 6x_8 = 7\\
x_6 &- 6x_7 + 7x_8 = 7; \qquad x_i \geqslant 0, \qquad i = 1, 2, \ldots, 8.
\end{aligned}$$

O bază se poate construi cu primele şase coloane, iar soluţia de pornire este

$$\mathbf{x}^1 = (2, 5, 14, 9, 7, 7, 0, 0).$$

Deoarece

$$\frac{\partial f}{\partial x_1} = \frac{(1 - \sqrt{1 + x_1x_2})^3}{x_1^3\sqrt{1 + x_1x_2}}, \quad \frac{\partial f}{\partial x_2} = -\frac{1 - \sqrt{1 + x_1x_2}}{x_1\sqrt{1 + x_1x_2}}$$

$$\frac{\partial f}{\partial x_i} = 0, \qquad (i = 3, 4, \ldots, 8), \text{ avem}$$

$$c_1 = \frac{\partial f(\mathbf{x}^1)}{\partial x_1} = -0{,}46, \quad c_2 = \frac{\partial f(\mathbf{x}^1)}{\partial x_2} = 0{,}35, \quad c_i = 0, \qquad (i = 3, 4, \ldots, 8).$$

Calculînd

$$c_h - \sum_{j=1}^{6} c_j\alpha_{jh}, \qquad (h = 1, 2, \ldots, 8),$$

se constată că unul singur este pozitiv:

$$c_8 - \sum_{j=1}^{6} c_j\alpha_{j8} \simeq 1{,}2.$$

Prin urmare se va introduce în bază vectorul $\mathbf{k}_8$. Noul punct (plan) va fi

$$\mathbf{x}^2 = (1, 7, 23, 5, 1, 0, 0, 1).$$

Scanned with OKEN Scanner

Acest punct însă nu este optimal. Introducînd în bază în locul lui vectorul $\mathbf{k}_7$ se obţine

$$\mathbf{x}^3 = (2,\ 12,\ 42,\ 2,\ 0,\ 0,\ 7,\ 7)$$

care este deja un punct optimal.

Dacă în ultima bază se înlocuieşte vectorul $\mathbf{k}_4$ cu $\mathbf{k}_6$ se obţine un nou punct optimal

$$\mathbf{x}^4 = (3,\ 16,\ 57,\ 0,\ 0,\ 1,\ 13,\ 12)$$

şi $f(\mathbf{x}^3) = f(\mathbf{x}^4) = 4$. Din monotonia lui f rezultă că $\mathbf{x}^* = \lambda\mathbf{x}^3 + (1 - \lambda)\mathbf{x}^4$, $0 \leqslant \lambda \leqslant 1$ este tot un punct optimal. De altfel aceste puncte formează domeniul Ω^*.

Scanned with OKEN Scanner

CAPITOLUL IV

PROGRAMAREA PĂTRATICĂ

§ 18. Algoritmul lui Beale-Zuhoviţki

Din punct de vedere istoric, prima problemă de programare neliniară a fost aceea cunoscută astăzi sub denumirea de programare pătratică. Prin problemă de programare pătratică se înţelege o problemă de programare matematică în care funcţia de scop este pătratică convexă, iar restricţiile sînt liniare. Într-un sens mai larg se poate înţelege printr-o problemă de programare pătratică şi o problemă de programare în care funcţia de scop este pătratică, dar nu este neapărat convexă. Este adevărat că în marea majoritate a metodelor cunoscute se cere ca funcţia să fie convexă (chiar strict convexă).

Multe metode cunoscute pentru programarea pătratică se bazează pe o adaptare convenabilă a metodelor gradientului, valabile şi în cazul general al unei programări convexe oarecare, cu restricţii liniare (vezi KÜNZI, P. H. şi KRELLE, W. [42] şi WOLFE, PH. [79]).

Problema de programare pătratică a stîrnit un interes deosebit în rîndul matematicienilor, datorită, pe de o parte, faptului că aceasta este o primă extensiune a problemei de programare liniară, iar, pe de altă parte, pentru faptul că în acest caz s-a putut construi un algoritm finit pentru rezolvarea problemei. Totuşi, în practică, această problemă este insuficient de intensiv utilizată. Se pare că acest lucru se datoreşte următoarelor trei cauze: 1) În practică sînt mult mai importante problemele cu restricţii neliniare decît cele cu restricţii liniare şi funcţia de scop neliniară. 2) Cînd apare o problemă de programare cu funcţia de scop neliniară, atunci de obicei doar un număr relativ mic de variabile apar neliniar, iar restul liniar şi de aceea nu sînt prea avantajoase metodele construite pentru problema de programare pătratică convexă. 3) De cele mai multe ori funcţia de scop care apare nu este convexă. Or, metodele folosite se aplică numai pentru funcţii convexe.

Algoritmul finit pentru problema de programare pătratică a fost dat pentru prima dată în anul 1955 de către BEALE, E. M. L. [1], iar ulterior, în anul 1962, de către ZUHOVIŢKI S. I. şi LEIFMAN, L. I. [84], WOLFE PH. [78] a dat o variantă în 1963.

Vom descrie în cele ce urmează varianta dată de ZUHOVIŢKI ([86], pag. 329).

Scanned with OKEN Scanner

Să considerăm deci problema de programare pătratică: *să se minimizeze funcţia*:

$$f(\mathbf{x}) = \frac{1}{2}\,\mathbf{x}'\mathbf{C}\mathbf{x} + \mathbf{c}\mathbf{x} \tag{18.1}$$

în condiţiile

$$\mathbf{A}\mathbf{x} \leqslant \mathbf{b}, \tag{18.2}$$

unde matricea $\mathbf{C}$ *este o matrice pătratică simetrică de tipul* (n, n), $\mathbf{A}$ *este o matrice de tipul* (m, n), *iar* $\mathbf{b}$ *un vector coloană de tipul* $(1, m)$. Pentru ca funcţia $f(\mathbf{x})$ să fie convexă, se presupune că $\mathbf{C}$ este o matrice *pozitiv-semi definită.*

1) Se construieşte un tabel simplex cu coeficienţii matricei $\mathbf{A}$, a matricei $\mathbf{C}$ şi a termenilor liberi, adică un tabel care să conţină coeficienţii restricţiilor şi coeficienţii derivatelor parţiale ale funcţiei de scop.

	$x_1 \ldots x_j \ldots x_n$	1
$g_1 =$	$a_{11} \ldots a_{1j} \ldots a_{1n}$	b_1
$\ldots$	$\ldots\ldots\ldots$	$\ldots$
$g_i =$	$a_{i1} \ldots a_{ij} \ldots a_{in}$	b_i
$\ldots$	$\ldots\ldots\ldots$	$\ldots$
$g_m =$	$a_{m1} \ldots a_{mj} \ldots a_{mn}$	b_m
$f'_{x_1} =$	$c_{11} \ldots c_{1j} \ldots c_{1n}$	c_1
$\ldots$	$\ldots\ldots\ldots$	$\ldots$
$f'_{x_k} =$	$c_{k1} \ldots c_{kj} \ldots c_{kn}$	c_k
$\ldots$	$\ldots\ldots\ldots$	$\ldots$
$f'_{x_n} =$	$c_{n1} \ldots c_{nj} \ldots c_{nn}$	c_n

Determinăm mai întîi punctul α^0, în care funcţia $f(\mathbf{x})$ îşi atinge minimul. Dacă $\alpha^0 \in \Omega$, atunci $\alpha^0 = \mathbf{x}^*$ şi problema este rezolvată, deoarece atunci punctul $\mathbf{x}^*$ de minim relativ este şi punctul în care funcţia $f(\mathbf{x})$ îşi atinge minimul absolut în domeniul Ω. Dacă însă $\alpha^0 \notin \Omega$, atunci alegem un punct arbitrar din domeniul Ω, $\mathbf{x}^0 \in \Omega$ (ca punct de pornire) şi ne deplasăm de-a lungul dreptei determinate de vectorul $\mathbf{x}^1 = \alpha^0 - \mathbf{x}^0$, pînă ce întîlnim frontiera domeniului într-un punct $\mathbf{x}^1$, pe care îl vom considera ca prima aproximaţie succesivă. După cum am văzut deja, acest punct se obţine din

$$\mathbf{x} = \mathbf{x}^0 + t\mathbf{s}^1$$

luînd pentru t valoarea minimă t_1 a rapoartelor pozitive

$$\frac{-\,g_1(\mathbf{x}^0)}{\mathbf{a}_1\mathbf{s}^1}, \ldots, \frac{-\,g_m(\mathbf{x}^0)}{\mathbf{a}_m\mathbf{s}^1}, \; 1.$$

Scanned with OKEN Scanner

Să presupunem că în punctul $\mathbf{x}^1$ avem

$$g_1(\mathbf{x}^1) = \ldots = g_p(\mathbf{x}^1) = 0. \tag{18.3}$$

Efectuăm apoi toţi paşii de eliminare Jordan (vezi ZUHOVIŢKI [86,] pag. 10) prin care aducem deasupra tabelului o parte din funcţiile g_j. În această eliminare, fiecare pas de eliminare Jordan se completează cu următoarea operaţie (care realizează regula derivării unei funcţii compuse): dacă la un pas Jordan se schimbă locul între variabila x_j şi funcţia g_i(adică a_{ij} este elementul pivot), atunci în tabelul obţinut la elementele fiecărei linii f'_{x_k}, pentru $k \neq j$, se adună elementele corespunzătoare din linia f'_{x_j} înmulţite cu noua valoare a elementului a_{ik} $\left(\text{adică cu } -\frac{a_{ik}}{a_{ij}}\right)$. Liniile obţinute se notează din nou cu aceleaşi litere f'_{x_k}. Elementele noii linii f'_{x_j} se înmulţesc numai cu noua valoare a elementului a_{ij} (adică cu $1/a_{ij}$) şi linia se notează cu f'_{g_i}.

Această regulă de construire a noului tabel simplex rezultă dacă se scoate variabila x_j din relaţia $g_i(\mathbf{x}) = 0$, se înlocuieşte în celelalte funcţii şi apoi se derivează f ca o funcţie compusă în raport cu x_k.

În cele ce urmează vom înţelege printr-un pas Jordan, un pas Jordan obişnuit cu modificările amintite.

După $r (r \leqslant p)$ paşi Jordan, tabelul simplex iniţial se va transforma într-un tabel de forma:

(18.4)

	g_1	$\ldots$	g_r	x_{r+1}	$\ldots$	x_n	1
$g_{r+1} =$	$a^r_{r+1,1}$	$\ldots$	$a^r_{r+1,r}$	0	$\ldots$	0	0
$\ldots$	$\ldots$						$\ldots$
$g_p =$	a^r_{p1}	$\ldots$	a^r_{pr}	0	$\ldots$	0	0
$g_{p+1} =$	$a^r_{p+1,1}$	$\ldots$	$a^r_{p+1,r}$	$a^r_{p+1,r+1}$	$\ldots$	$a^r_{p+1,n}$	b^r_{p+1}
$\ldots$	$\ldots$						$\ldots$
$g_m =$	a^r_{m1}	$\ldots$	a^r_{mr}	$a^r_{m,r+1}$	$\ldots$	a^r_{mn}	b^r_m
$f'_{g_1} =$	c^r_{11}	$\ldots$	c_{1r}	$c^r_{1,r+1}$	$\ldots$	c_{1n}	c^r_1
$\ldots$	$\ldots$						$\ldots$
$f'_{g_r} =$	c^r_{r1}	$\ldots$	c^r_{rr}	$c^r_{r,r+1}$	$\ldots$	c^r_{rn}	c^r_r
$f'_{x_{r+1}} =$	$c^r_{r+1,1}$	$\ldots$	$c^r_{r+1,r}$	$c^r_{r+1,r+1}$	$\ldots$	$c^r_{r+1,n}$	c^r_{r+1}
$\ldots$	$\ldots$						$\ldots$
$f'_{x_n} =$	c^r_{n1}	$\ldots$	c^r_{nr}	$c^r_{n,r+1}$	$\ldots$	c^r_{nn}	c^r_n

Scanned with OKEN Scanner

2) Se găseşte minimul funcţiei de scop $f(\mathbf{x})$ cu legăturile (18.3). Fie α^2 punctul de minim relativ obţinut. Pentru aceasta se rezolvă sistemul.

$$c^r_{j,r+1}x_{r+1} + \ldots + c^r_{jn}x_n + c^r_j = 0, \quad (j = r+1, \ldots, n).$$

Noua direcţie de scufundare va fi $\mathbf{s}^2 = \alpha^2 - \mathbf{x}^1$. Ne deplasăm de-a lungul acestei direcţii, cu punctele de forma

$$\mathbf{x} = \mathbf{x}^1 + t\mathbf{s}^2$$

pînă ce vom întîlni pentru prima dată o suprafaţă de restricţie care nu trece prin $\mathbf{x}^1$, adică luînd pentru parametrul t valoarea t_2 egală cu cel mai mic număr pozitiv dintre numerele

$$\frac{g_{p+1}(\mathbf{x}^1)}{\mathbf{a}_{p+1}\mathbf{s}^2}, \ \ldots, \ \frac{-g_m(\mathbf{x}^1)}{\mathbf{a}_m\mathbf{s}^2}, \ 1.$$

Dacă $t = 1$, atunci $\alpha^2 \in \Omega$ şi $\mathbf{x}^2 = \alpha^2$. Un asemenea punct se va numi (după Zuhoviţki) *punct staţionar*.

Dacă $p = 1$, atunci punctul staţionar α^2 este soluţia problemei.

Dacă $p > 1$, atunci punctul staţionar s-ar putea să nu fie punct optimal, de aceea se va căuta o nouă direcţie de scufundare, ca la punctul 3).

Dacă $0 < t_2 < 1$, atunci după cel mult $n - r$ paşi, ori obţinem un punct staţionar, ori ajungem la vîrf, adică pentru care $p \geqslant m$, care se va numi tot punct staţionar.

3) Vom determina direcţia de scufundare dintr-un punct staţionar în felul următor: Fie punctul staţionar

$$\tilde{\mathbf{x}}(0, \ldots, 0, \ x_{r+1}, \ \ldots, \ x_n)$$

care aparţine hiperplanelor

$$g_1(\mathbf{x}) = 0, \ \ldots, g_r(\mathbf{x}) = 0, \ g_{r+1}(\mathbf{x}) = 0, \ \ldots, g_p(\mathbf{x}) = 0 \tag{18.5}$$

(vezi tabelul (18.4)). Atunci dacă $p = 1$, punctul $\tilde{\mathbf{x}}$ este soluţia căutată.

Dacă $p > 1$, atunci pentru găsirea direcţiei celei mai rapide scufundări se rezolvă problema: să se minimizeze funcţia

$$u = \sum_{i=1}^{r} \frac{\partial f(\tilde{\mathbf{x}})}{\partial g_i} s_i$$

în condiţiile

$$-C \leqslant s_i \leqslant 0, \qquad (i = 1, 2, \ldots, r)$$

$$\frac{\partial g_i}{\partial \mathbf{s}} = a^r_{j1}s_1 + \ldots + a^r_{jr}s_r \leqslant 0, \qquad (j = r+1, \ldots, p)$$

(deoarece $s_{r+1}, \ldots, s_n$ nu participă în programarea noastră liniară, ele se pot lua egale cu zero).

Scanned with OKEN Scanner

Dacă $\min u = 0$, atunci $\mathbf{x}^* = \tilde{\mathbf{x}}$, şi problema este rezolvată. Dacă $\min u < 0$, atunci ne deplasăm din punctul $\tilde{\mathbf{x}}$ pe direcţia găsită $\tilde{\mathbf{s}}$ considerînd punctele de forma

$$\mathbf{x} = \tilde{\mathbf{x}} + t\tilde{\mathbf{s}}$$

şi calculăm valoarea lui t care corespunde primului plan întîlnit sau minimului funcţiei $f(\mathbf{x} + t\mathbf{s})$. Cu punctul astfel obţinut se procedează apoi ca la punctul 1). După un anumit număr de paşi Jordan se va ajunge la un tabel analog cu (18.4), de la care se continuă ca la punctul 2) pînă se ajunge la un nou punct staţionar ş.a.m.d.

Deoarece algoritmul este monoton, iar numărul punctelor staţionare este finit, rezultă că *după un număr finit de paşi se va ajunge la soluţia problemei.*

Exemplul 18.1. Să se minimizeze funcţia

$$f(\mathbf{x}) = 2x_1^2 + x_2^2 + 3x_3^2 - x_1x_2 - 2x_1x_3 - x_2x_3 + 10x_1 + 9x_2 - 26x_3$$

supusă restricţiilor

$$\begin{aligned}
g_1(\mathbf{x}) &= -x_1 - 1 \leqslant 0\\
g_2(\mathbf{x}) &= -x_2 \leqslant 0\\
g_3(\mathbf{x}) &= -x_3 \leqslant 0\\
g_4(\mathbf{x}) &= x_1 + x_2 + x_3 - 4 \leqslant 0\\
g_5(\mathbf{x}) &= -x_1 - 2x_2 - x_3 + 3 \leqslant 0
\end{aligned}$$

Tabelul simplex corespunzător este următorul:

(18.6)

	x_1	x_2	x_3	1
$g_1 =$	−1	0	0	−1
$g_2 =$	0	−1	0	0
$g_3 =$	0	0	−1	0
$g_4 =$	1	1	1	−4
$g_5 =$	−1	−2	**[−1]**	3
$f'_{x_1} =$	4	−1	−2	10
$f'_{x_2} =$	−1	2	−1	9
$f'_{x_3} =$	−2	−1	6	−26

Scanned with OKEN Scanner

Pasul 1. 1) *Determinarea punctului de minim.* Se rezolvă sistemul

$$f'_{x_1} = 4x_1 - x_2 - 2x_3 + 10 = 0$$

$$f'_{x_2} = -x_1 + 2x_2 - x_3 + 9 = 0$$

$$f'_{x_3} = -2x_1 - x_2 + 6x_3 - 26 = 0$$

şi găsim soluţia

$$\alpha^0 = (-2,\ -4,\ 3) \notin \Omega.$$

Considerăm un punct oarecare al domeniului Ω. Se vede că $\mathbf{x}^0 = (1, 1, 1,) \in \Omega$, căci $g_i(\mathbf{x}^0) < 0$, $i = 1, 2, \ldots, 5$.

2) *Determinarea direcţiei de scufundare* $\mathbf{s}^1$. În acest caz

$$\mathbf{s}^1 = \alpha^0 - \mathbf{x}^0 = (-3,\ -5,\ 2).$$

3) *Determinarea pasului* t_1. Deplasîndu-ne pe semidreapta

$$x_1 = 1 - 3t,\ x_2 = 1 - 5t,\ x_3 = 1 + 2t,\ t > 0,$$

obţinem pentru t valoarea

$$t_1 = \frac{1}{11}$$

care corespunde punctului frontieră întîlnit.

4) *Determinarea noului punct de aproximaţie succesivă.* Avem:

$$\mathbf{x}^1 = \mathbf{x}^0 + t_1\mathbf{s}^1 = \left(\frac{8}{11},\ \frac{6}{11},\ \frac{13}{11}\right),$$

$$g_5(\mathbf{x}^1) = 0,\ g_i(\mathbf{x}^1) < 0,\ (i \neq 5).$$

Pasul 2. 1) *Determinarea punctului de minim legat.* Efectuăm un pas Jordan descris mai sus în Tabelul (18.6) cu elementul pivot marcat după care se va ajunge la tabelul

(18.7)

	x_1	x_2	g_5	1
$g_1 =$	−1	0	0	−1
$g_2 =$	0	$\boxed{-1}$	0	0
$g_3 =$	1	2	1	−3
$g_4 =$	0	−1	−1	−1
$f'_{x_1} =$	14	16	8	12
$f'_{x_2} =$	16	30	13	22
$f'_{g_5} =$	8	13	6	8

Scanned with OKEN Scanner

şi

$$x_3 = -x_1 - 2x_2 - g_5 + 3.$$

Se rezolvă sistemul

$$(f'_{x_1})_{g_5=0} = 14x_1 + 16x_2 + 12 = 0$$

$$(f'_{x_2})_{g_5=0} = 16x_1 + 30x_2 + 22 = 0$$

care conduce la soluţia

$$\alpha^1 = \left(-\frac{2}{41}, -\frac{29}{41}, 0\right) \notin \Omega.$$

2) *Determinarea direcţiei* s^2. Avem

$$s^2 = \alpha^1 - x^1 = \left(-\frac{350}{451}, -\frac{565}{451}, 0\right).$$

3) *Determinarea pasului* t_2. Deplasîndu-ne pe semidreapta $x = x^1 + ts^2$, adică

$$x_1 = \frac{8}{11} - \frac{350}{451}t$$

$$x_2 = \frac{6}{11} - \frac{565}{451}t$$

$$g_5 = 0,$$

obţinem

$$t_2 = \frac{246}{565}.$$

4) *Determinarea noului punct de aproximaţie succesivă.* Avem

$$x^2 = \left(-\frac{44}{113}, 0, 0\right)$$

şi

$$g_2(x^2) = 0, \quad g_5(x^2) = 0, \quad g_i(x^2) < 0, \quad i = 1, 3, 4.$$

Pasul 3. 1) *Determinarea punctului de minim cu legături.* Se efectuează un pas Jordan analog cu cel precedent cu elementul pivot încadrat în Tabelul (18.7). Se va ajunge la tabelul

	x_1	g_2	g_5	1
$g_1 =$	-1	0	0	-1
$g_3 =$	1	-2	1	-3
$g_4 =$	0	1	-1	-1
$f'_{x_1} =$	14	-16	8	12
$f'_{g_2} =$	-16	30	-13	-22
$f'_{g_5} =$	8	-13	6	8

şi

$$x_2 = -g_2.$$

Soluţia ecuaţiei

$$(f'_{x_1})_{g_2=g_5=0} = 14x_1 + 12 = 0.$$

este $\alpha^2 = (x_2, g_2, g_5) = \left(-\frac{6}{7}, 0, 0\right) \in \Omega$. Prin urmare punctul $\mathbf{x}^3 = \alpha^3$ este un punct staţionar.

Pasul 4. Se rezolvă următoarea problemă de programare liniară: să se minimizeze funcţia

$$u = \frac{\partial f(\mathbf{x}^3)}{\partial g_2} s_2 + \frac{\partial f(\mathbf{x}^3)}{\partial g_3} s_3 = -8\frac{2}{7} s_2 + 1\frac{1}{7} s_3$$

supusă restricţiilor

$$s_1 = 0$$
$$-1 \leqslant s_2 \leqslant 0.$$
$$-1 \leqslant s_3 \leqslant 0.$$

Soluţia acestei probleme este $\mathbf{s}^4 = (0, 0, -1)$ şi $u_{min} = -1\frac{1}{7} < 0$

3) *Determinarea pasului* t_4. Deplasîndu-ne pe semidreapta $\mathbf{x} = \mathbf{x}^3 + t\mathbf{s}^4$ pentru $t > 0$, obţinem $t_4 = t' = \frac{4}{21}$, unde t' este valoarea lui t pentru care funcţia $f(\mathbf{x}^3 + t\mathbf{s}^4)$ ia valoarea minimă.

Scanned with OKEN Scanner

4) *Determinarea noului punct de aproximație succesivă.* Avem:

$$\mathbf{x}^4 = \mathbf{x}^3 + t_4 \mathbf{s}^4 = \left(-\frac{6}{7},\ 0,\ -\frac{4}{21}\right) \in \Omega,$$

$$g_i(\mathbf{x}^4) < 0,\ (i = 1,\ 3,\ 4,\ 5).$$

Pasul 5. Se determină punctul de minim cu legături a funcției de scop rezolvînd sistemul

$$(f'_{x_1})_{g_2=0} = 14x_1 + 8g_5 + 12 = 0$$

$$(f'_{g_5})_{g_2=0} = 8x_1 + 6g_5 + 8 = 0$$

și se obține

$$\alpha^4 = (-0{,}4;\ 0;\ -0{,}8) \in \Omega,$$

așa că $\mathbf{x}^4 = \alpha^4$ este din nou un punct staționar. Deoarece $p = 1$, rezultă că $\mathbf{x}^4 = (-0{,}4;\ 0;\ -0{,}8)$ sau în vechile coordonate $\mathbf{x}^4 = (-0{,}4;\ 0;\ 4{,}2)$ este soluția problemei.

§ 19. Reprezentarea funcției de scop ca sumă de pătrate

Multe metode care se utilizează pentru rezolvarea problemei de programare pătratică se simplifică simțitor dacă funcția de scop este dată sub forma unei sume de pătrate. BEALE E. M. [1] propune o metodă directă destul de ușoară care permite scrierea oricărei funcții pătratice sub forma unei sume de pătrate.

Astfel, să considerăm funcția

$$f(\mathbf{x}) = \sum_{i=1}^{n-m} \sum_{k=1}^{n-m} c_{ik} x_i x_k,$$

unde $c_{ik} = c_{ki}$. Vom urmări cel mai mare coeficient în valoare absolută dintre coeficienții c_{ik}.

1°. Dacă acesta este un coeficient c_{pp} de pe diagonală, atunci se procedează astfel: dacă $c_{pp} > 0$, atunci se pune

$$u_1 = \sqrt{\frac{2}{c_{pp}} \sum_{k=1}^{n-m} c_{kp} x_k}.$$

Scanned with OKEN Scanner

Avem:

$$f(\mathbf{x}) = \frac{1}{2}u_1^2 + \sum_{i=1}^{n-m}{}^{(p)} \sum_{k=1}^{n-m}{}^{(p)} c'_{ik}x_i x_k, \tag{19.1}$$

unde

$$c'_{ik} = c_{ik} - c_{ip}c_{pk}/c_{pp}. \tag{19.2}$$

În mod analog se procedează dacă c_{pp} este negativ, punînd

$$u_1 = \sqrt{\frac{-2}{c_{pp}} \sum_{k=1}^{n-m} c_{kp}x_k}$$

şi funcţia $f(\mathbf{x})$ se va scrie

$$f(\mathbf{x}) = -\frac{u_1^2}{2} + \Sigma\, c'_{ik}x_i x_k$$

unde c'_{ik} se definesc după formula (19.2).

Acum se poate repeta procedeul cu suma rămasă.

2°. Coeficientul cel mai mare în valoare absolută nu este pe diagonală. Fie acesta c_{pq}, $p \neq q$. În acest caz se face o schimbare prealabilă de variabile:

$$y_p = x_p + x_q$$

$$y_q = x_p - x_q.$$

Atunci

$$f(\mathbf{x}) = \sum_{i=1}^{n-m} \sum_{k=1}^{n-m} c_{ik}x_i x_k = \sum_{i=1}^{n-m} \sum_{k=1}^{n-m} c'_{ik}y_i y_k$$

unde $y_k = x_k$, pentru $k \neq p, q$.

Se poate verifica că dacă c_{pq} este coeficientul cel mai mare în valoare absolută printre coeficienţii c_{ik}, atunci unul din coeficienţii c'_{pp} sau c'_{qq} trebuie să fie cel mai mare în valoare absolută. Astfel sîntem din nou în situaţia de la 1°. Scriind funcţia f sub forma (19.1) se revine la variabilele x prin transformarea inversă.

Cunoscînd acest procedeu se va ajunge la următoarea formă a funcţiei de scop:

$$f(\mathbf{x}) = u_0 + \frac{1}{2}\sum_{i=1}^{r_1} u_i^2 - \frac{1}{2}\sum_{i=r_1+1}^{r} u_i^2, \tag{19.3}$$

unde u_i sînt funcţii liniare de variabilele x_k. Funcţia $f(\mathbf{x})$ este convexă dacă $r_1 = r$ şi este concavă dacă $r_1 = 0$.

Scanned with OKEN Scanner

§ 20. Metoda funcţiilor omogene în rezolvarea problemelor de programare pătratică

Folosind proprietăţile funcţiilor omogene, MOCANU C. [59], [60] a indicat o transformare care permite reducerea unei probleme de programare pătratică convexe oarecare la o formă canonică care se pretează mult mai uşor algoritmizării. Astfel, dacă se consideră o funcţie f omogenă de ordinul α şi funcţiile g_j omogene de ordinul α_i, atunci considerînd problema de extremum cu legături a funcţiei f supusă legăturilor

$$g_j(\mathbf{x}) = a_j, \qquad (j = 1, 2, \ldots, m) \tag{20.1}$$

$(m < n)$ şi folosind metoda multiplicatorilor lui Lagrange, se ajunge la funcţia

$$F(\mathbf{x}, \lambda) = f(\mathbf{x}) - \sum_{j=1}^{m} \lambda_j [g_j(\mathbf{x}) - a_j]. \tag{20.2}$$

Vectorul soluţiei $\mathbf{x}$ şi vectorul $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ al multiplicatorilor trebuie să satisfacă condiţiile

$$\frac{\partial f}{\partial x_i} - \sum_{j=1}^{m} \lambda_j \frac{\partial g_j}{\partial x_i} = 0, \qquad (i = 1, 2, \ldots, n), \tag{20.3}$$

$$g_j(\mathbf{x}) = a_j.$$

Dacă înmulţim primele n ecuaţii din (20.3) cu x_i şi însumăm membru cu membru, ţinînd seama de identitatea lui Euler pentru funcţii omogene, obţinem

$$\alpha f(\mathbf{x}) - \sum_{j=1}^{m} \alpha_j \lambda_j g_j(\mathbf{x}) = 0. \tag{20.4}$$

Dacă $\tilde{\mathbf{x}}$ este soluţia sistemului (20.3), atunci

$$g_j(\tilde{\mathbf{x}}) = a_j$$

şi rezultă

$$\alpha f(\tilde{\mathbf{x}}) - \sum_{j=1}^{m} \alpha_j \lambda_j a_j = 0.$$

Prin urmare, dacă $\alpha \neq 0$ atunci valoarea funcţiei f în punctul de extremum este dată de egalitatea

$$f(\tilde{\mathbf{x}}) = \frac{1}{\alpha} \sum_{j=1}^{m} \alpha_j \lambda_j a_j. \tag{20.5}$$

Scanned with OKEN Scanner

Considerăm acum problema de programare pătratică: *să se minimizeze funcţia*

$$f(\mathbf{x}) = \frac{1}{2}\,\mathbf{x}'\mathbf{C}\mathbf{x} \tag{20.6}$$

în condiţiile

$$\mathbf{A}\mathbf{x} \geqslant \mathbf{b}, \tag{20.7}$$

unde $\mathbf{C}$ *este o matrice simetrică pozitiv-definită.*

Punînd

$$\mathbf{A}\mathbf{x} = \mathbf{u},\ \mathbf{u} \geqslant \mathbf{b},$$

problema enunţată se înlocuieşte cu următoarea: *să se minimizeze funcţia* $f(\mathbf{x})$ *din* (20.6) *în condiţiile*

$$\mathbf{A}\mathbf{x} = \mathbf{u}$$

unde $\mathbf{u}$ *este un vector dat.*

Întrucît f şi g_j sînt funcţii omogene, aplicînd egalitatea (20.5), obţinem

$$\min f(\mathbf{x}) = f(\tilde{\mathbf{x}}) = \frac{1}{2}\sum_{i=1}^{m}\lambda_j u_j = \frac{1}{2}\,\mathbf{u}'\lambda. \tag{20.8}$$

Parametrul λ se poate elimina, dacă se utilizează convenabil relaţiile (20.3), care dau condiţiile necesare de extremum. Aceste condiţii se pot scrie şi sub forma

$$\nabla f(\mathbf{x}) - \nabla \mathbf{G}(\mathbf{x})\lambda = \mathbf{0},$$

care pentru $f(\mathbf{x}) = 1/2\mathbf{x}'\mathbf{C}\mathbf{x}$ şi $\mathbf{G}(\mathbf{x}) = \mathbf{A}\mathbf{x}$, conduc la egalitatea

$$\mathbf{C}\mathbf{x} = \mathbf{A}'\lambda.$$

De aici rezultă

$$\tilde{\mathbf{x}} = \mathbf{C}^{-1}\mathbf{A}'\lambda.$$

Înlocuind această expresie în $\mathbf{A}\mathbf{x} = \mathbf{u}$, avem

$$(\mathbf{A}\mathbf{C}^{-1})\mathbf{A}'\lambda = \mathbf{u},$$

Scanned with OKEN Scanner

deci

$$\lambda = (\mathbf{AC}^{-1}\mathbf{A}')^{-1}\,\mathbf{u} = \mathbf{Ku},$$

unde $\mathbf{K} = (\mathbf{AC}^{-1}\mathbf{A}')^{-1}$.

Revenind la valoarea minimă a funcţiei de scop dată în (20.8), avem

$$f(\mathbf{u}) = \frac{1}{2}\,\mathbf{u}'\mathbf{Ku}.$$

Astfel, problema de programare pătratică (20.6)—(20.7) este echivalentă cu următoarea problemă: *să se minimizeze funcţia*

(20.9) $$f(\mathbf{u}) = \frac{1}{2}\,\mathbf{u}'\mathbf{Ku}$$

n condiţiile

20.10) $$\mathbf{u} \geqslant \mathbf{b}.$$

Această problemă este mai uşoară, deoarece restricţiile sînt mult mai simple.

Pentru această problemă se dă un algoritm destul de simplu cu care problema se rezolvă într-un număr finit de paşi [59].

Exemplul 20.1. Să se determine minimul funcţiei

$$f(\mathbf{x}) = (x_1 - x_2)^2 + (x_2 - x_3)^2 + (x_3 - x_4)^2 + x_4^2$$

supusă restricţiilor

$$\begin{aligned} x_1 \quad + x_3 \quad &\geqslant 1 \\ x_2 \quad + x_4 \quad &\geqslant 1 \\ 2x_1 + x_2 \quad &\geqslant 2 \\ x_3 + 2x_4 \quad &\geqslant 3. \end{aligned}$$

Forma pătratică $f(\mathbf{x})$ este evident pozitiv definită. Vom transforma această problemă într-una de tipul (20.9)—(20.10). Avem

$$\mathbf{C} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{pmatrix}; \quad \mathbf{A} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 \end{pmatrix}$$

Scanned with OKEN Scanner

$$\mathbf{AC^{-1}A'} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} 4 & 3 & 2 & 1 \\ 3 & 3 & 2 & 1 \\ 2 & 2 & 2 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 2 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \end{pmatrix} =$$

$$= \begin{pmatrix} 10 & 7 & 17 & 8 \\ 7 & 6 & 12 & 7 \\ 17 & 12 & 31 & 12 \\ 8 & 7 & 12 & 10 \end{pmatrix}.$$

$$\mathbf{K} = (\mathbf{AC^{-1}A'})^{-1} = \frac{1}{9} \begin{pmatrix} 53 & 34 & -31 & -29 \\ 34 & 50 & -26 & -31 \\ -31 & -26 & 20 & 19 \\ -29 & -31 & 19 & 23 \end{pmatrix}$$

Astfel noua problemă de programare pătratică este următoarea: să se minimizeze funcţia:

$$f(\mathbf{u}) = \frac{1}{18}(53u_1^2 + 50u_2^2 + 20u_3^2 + 23u_4^2 + 68u_1u_2 - 62u_1u_3 -$$

$$-58u_1u_4 - 52u_2u_3 - 62u_2u_4 + 38u_3u_4).$$

supusă restricţiilor

$$u_1 \geqslant 1,\ u_2 \geqslant 4,\ u_3 \geqslant 2,\ u_4 \geqslant 3.$$

§ 21. O metodă a ortogonalizării

În anul 1965 MARUŞCIAC I. şi RĂDULEŞCU M. [51] au elaborat un algoritm pentru rezolvarea problemei de programare pătratică care prezintă o serie de avantaje faţă de alte metode, atît în ce priveşte calculul manual, cît şi elaborarea programului pentru maşină.

Procedeul rezultă dintr-o îmbinare a metodei gradientului proiectat şi a metodei ortogonalizării, cunoscută în teoria ecuaţiilor liniare.

În această metodă se evită atît rezolvarea unei probleme de programare liniară la fiecare pas, cît şi inversarea unor matrici. Calculele necesare sînt aşezate într-un tabel, analog cu tabelul simplex, care uşurează folosirea metodei şi de către nespecialişti.

Astfel, se consideră problema următoare: *să se minimizeze funcţia*

(21.1) $$f(\mathbf{x}) = \mathbf{x'Cx} + \mathbf{dx}$$

Scanned with OKEN Scanner

în condiţiile

$$\mathbf{Ax} + \mathbf{b} \geqslant \mathbf{0}, \tag{21.2}$$

unde $\mathbf{A}$ *este o matrice de tipul* (m, n), $\mathbf{C}$ — *o matrice pătratică de tipul* (n, n) *simetrică şi pozitiv definită, iar* $\mathbf{b}$ şi $\mathbf{d}$ *sînt vectori respectiv de tipul* $(m, 1)$ *şi* $(1, n)$.

Algoritmul începe cu un punct arbitrar $\mathbf{x}^0$, din domeniul admisibil Ω. Din acest punct ne deplasăm de-a lungul gradientului $\mathbf{s}^0 = \nabla f(\mathbf{x}^0)$, dacă aceasta este o direcţie admisibilă (dacă $\mathbf{x}^0 \in \operatorname{int} \Omega$, atunci $\nabla f(\mathbf{x}^0)$ este totdeauna o direcţie admisibilă). Verificarea dacă o direcţie $\mathbf{s}$ este admisibilă sau nu, se face în felul următor:

Dacă în punctul $\mathbf{x}^0$ avem

$$g_1(\mathbf{x}^0) = g_2(\mathbf{x}^0) = \ldots = g_p(\mathbf{x}^0) = 0, \; g_j(\mathbf{x}^0) > 0, \; j > p, \tag{21.3}$$

unde

$$g_j(\mathbf{x}) = \mathbf{a}_j\mathbf{x} + b_j,$$

atunci direcţia $\mathbf{s}$ este admisibilă dacă şi numai dacă

$$\mathbf{a}_j\mathbf{s} > 0, \qquad (j = 1, 2, \ldots, p). \tag{21.4}$$

Dacă direcţia $\mathbf{s}^0$ este admisibilă, atunci ne deplasăm pe dreapta

$$\mathbf{x} = \mathbf{x}^0 + t\mathbf{s}^0$$

pînă întîlnim un nou plan din (21.2) sau funcţia $f(\mathbf{x}^0 + t\mathbf{s}^0)$ îşi atinge valoarea minimă. Ca şi în varianta lui Zuhoviţki, noul punct de aproximaţie este

$$\mathbf{x}^1 = \mathbf{x}^0 + \tau\mathbf{s}^0,$$

unde τ este cea mai mică valoare a parametrului t astfel calculată.

Dacă direcţia gradientului lui f nu este admisibilă, atunci se consideră proiecţia sa pe planul $g_1(\mathbf{x}) = 0$ şi se verifică dacă aceasta este sau nu admisibilă. Dacă aceasta este admisibilă, atunci se continuă mai departe cu direcţia calculată. În caz contrar se proiectează gradientul lui f în punctul $\mathbf{x}^0$ pe planul $g_2(\mathbf{x}) = 0$ ş.a.m.d.

Dacă nici una din aceste direcţii nu este admisibilă, atunci se proiectează gradientul pe muchia comună a planelor (21.3) cu care se procedează la fel.

Scanned with OKEN Scanner

Proiecţiile gradientului se calculează în felul următor: Proiecţia gradientului funcţiei f în punctul $\mathbf{x}^0$ pe planul $g_i(\mathbf{x}) = 0$, este dată de

$$\mathbf{s}^{0,i} = \nabla f(\mathbf{x}^0) + \alpha_{0i}\mathbf{a}_i, \tag{21.5}$$

unde

$$\alpha_{0i} = \frac{-\nabla f(\mathbf{x}^0)\mathbf{a}_i}{\mathbf{a}_i\mathbf{a}_i'}.$$

Dacă notăm cu $\mathbf{s}^{0,1,2,\ldots,p}$ proiecţia gradientului $\mathbf{s}^0 = \nabla f(\mathbf{x}^0)$ pe planele $g_i(\mathbf{x}) = 0$ care trec prin punctul $\mathbf{x}^0$, atunci aceasta se calculează din aproape în aproape după formulele

$$\begin{aligned}
\mathbf{u}^1 &= \mathbf{a}_1 \\
\mathbf{u}^2 &= \mathbf{a}_2 + \alpha_{21}\mathbf{u}^1 \\
\mathbf{u}^3 &= \mathbf{a}_3 + \alpha_{31}\mathbf{u}^1 + \alpha_{32}\mathbf{u}^2 \\
&\ldots\ldots\ldots\ldots\ldots\ldots \\
\mathbf{u}^p &= \mathbf{a}_p + \alpha_{p1}\mathbf{u}^1 + \alpha_{p2}\mathbf{u}^2 + \ldots + \alpha_{p,p-1}\mathbf{u}^{p-1} \\
\mathbf{s}^{0,1,\ldots,p} &= \mathbf{s}^0 + \alpha_{01}\mathbf{u}^1 + \alpha_{02}\mathbf{u}^2 + \ldots + \alpha_{0p}\mathbf{u}^p,
\end{aligned} \tag{21.6}$$

unde

$$\alpha_{ik} = -\frac{\mathbf{a}_i\mathbf{u}^{k'}}{\mathbf{u}^k\mathbf{u}^{k'}},\ i > k,\ i \neq 0,\ \alpha_{0k} = -\frac{\mathbf{u}^0\mathbf{u}^{k'}}{\mathbf{u}^k\mathbf{u}^{k'}}.$$

Dacă pentru $p < n$ se ajunge la un vector $\mathbf{s}^{0,1,2,\ldots,p} = 0$, atunci punctul $\mathbf{x}^0$ este punctul optimal căutat.

Dacă $p \geqslant n$, atunci se proiectează pe toate muchiile unidimensionale care se pot forma din cele p plane, luate cîte $n - 1$. Dacă nici una dintre aceste proiecţii nu este admisibilă, atunci punctul este optimal.

În lucrare se arată apoi că şirul aproximaţiilor succesive $\mathbf{x}^k$ este convergent către punctul de optim căutat. De asemenea se arată că practic totdeauna se poate evita cazul cînd algoritmul conţine un număr infinit de paşi, rezolvînd o problemă de minim cu anumite legături, care sînt indicate de tabel.

Scanned with OKEN Scanner

§ 22. O formă explicită a soluţiei problemei de programare pătratică

Într-o lucrare destul de recentă a autorului [53], se prezintă un algoritm bazat pe exprimarea explicită a soluţiei minimului cu legături a funcţiei de scop. La această formă a soluţiei se ajunge folosind o metodă dată de Călugăreanu, G., în 1950, pentru exprimarea formei polinomului lui Cebîşev ataşat unei mulţimi finite din planul complex. Această metodă generalizată de autor în mod convenabil s-a dovedit extrem de eficace într-o serie de probleme din teoria aproximării funcţiilor.

Şi aici se consideră problema: *să se minimizeze funcţia de scop convexă*

$$f(\mathbf{x}) = \mathbf{x}'\mathbf{C}\mathbf{x} + \mathbf{b}\mathbf{x}' \tag{22.1}$$

în condiţiile

$$\mathbf{A}\mathbf{x} + \mathbf{d} \geqslant \mathbf{0}, \tag{22.2}$$

unde $\mathbf{C}$ *este o matrice simetrică pozitiv difinită,* $\mathbf{A}$ *o matrice de tipul* (m,n) *iar* $\mathbf{b}$ *şi* $\mathbf{d}$- *vectori respectiv de tipul* $(1, n)$ *şi* $(m,1)$.

Se pot întîmpla două cazuri: Soluţia optimă $\mathbf{x}^0 \in$ int (Ω), sau $\mathbf{x}^0 \in$ fr (Ω). În primul caz avem o problemă de programare pătratică banală, pentru că atunci soluţia problemei (22.1)—(22.2) nu depinde de restricţiile (22.2) şi se găseşte cu metoda obişnuită din analiză. În al doilea caz există o muchie de dimensiune $n-q$ a domeniului Ω, pe care este situat punctul $\mathbf{x}^0$, adică pentru care

$$g_{i_k}(\mathbf{x}^0) = 0, \qquad (k = 1,2, \ldots, q).$$

Pe de altă parte, deoarece funcţia

$$\varphi(\mathbf{x}) = \mathbf{x}'\mathbf{C}\mathbf{x}$$

este o funcţie convexă, ea se poate pune sub forma de sumă de pătrate (vezi § 19), adică sub forma

$$F(\mathbf{x}) = \sum_{k=1}^{n} c_k(a_{k1}x_1 + a_{k2}x_2 + \ldots + a_{kn}x_n)^2, \tag{22.3}$$

unde $c_k \geqslant 0$, $(k = 1,2, \ldots, n)$. Deci problema (22.1) — (22.2) se reduce la următoarea problemă: *să se minimizeze funcţia* $F(\mathbf{x})$ *în condiţiile*

$$g_j(\mathbf{x}) = \sum_{k=1}^{n} \alpha_{ik}x_k - \alpha_j = 0, \qquad (j = 1,2, \ldots, m).$$

Scanned with OKEN Scanner

Folosind metoda multiplicatorilor lui Lagrange, după mai multe transformări, se obţine soluţia problemei sub forma

$$x_k = \frac{\sum c_{\nu_1 \ldots c_{\nu_{n-m}}} D(\nu_1, \ldots, \nu_{n-m}; \alpha) D^{(k)}(\nu_1, \ldots, \nu_{n-m}; \alpha)}{\sum c_{\nu_1 \ldots c_{\nu_{n-m}}} [D(\nu_1, \ldots, \nu_{n-m}; \alpha)]^2}, \tag{22.4}$$

unde semnul Σ se ia pentru toţi indicii $\nu_1, \ldots, \nu_{n-m}$ de la 1 la n, iar

$$D(\nu_1, \ldots, \nu_{n-m}; \alpha) = \begin{vmatrix} a_{\nu_1 1} & \ldots & a_{\nu_1 n} \\ \ldots & \ldots & \ldots \\ a_{\nu_{n-m} 1} & \ldots & a_{\nu_{n-m} n} \\ \alpha_{11} & \ldots & \alpha_{1n} \\ \ldots & \ldots & \ldots \\ \alpha_{m1} & \ldots & \alpha_{mn} \end{vmatrix}$$

$D^{(k)}(\nu_1, \ldots, \nu_{n-m}; \alpha)$ fiind determinantul care se obţine din $D(\nu_1, \ldots, \nu_{n-m}; \alpha)$ punînd $a_{\nu_i k} = 0$ $(i = 1,2, \ldots, n-m)$ şi $\alpha_{i_k} = \alpha_i$ $(i = 1,2, \ldots, m)$.

Să ne întoarcem acum la problema de programare pătratică (22.3). Vom considera mai întîi cazul cînd $\operatorname{rang} \| \alpha_{ik} \| = m$ (adică $m \leqslant n$). Atunci algoritmul constă în următoarele: se calculează mai întîi minimul relativ $\mathbf{x}^1$ pentru $m = 1$, corespunzător legăturii

$$g_1(\mathbf{x}) = 0.$$

Dacă $\mathbf{x}^1 \in \Omega$, atunci $\mathbf{x}^1$ este soluţia optimă căutată. Dacă $\mathbf{x}^1 \notin \Omega$, atunci se calculează $\mathbf{x}^2$ pentru $m = 1$ şi legătura

$$g_2(\mathbf{x}) = 0,$$

cu care se procedează la fel.

Dacă nici unul din punctele $\mathbf{x}^k$ calculate pentru $m = 1$ nu aparţin domeniului Ω, atunci se calculează $\mathbf{x}^h$ corespunzător lui $m = 2$ şi legăturilor

$$g_1(\mathbf{x}) = g_2(\mathbf{x}) = 0$$

cu care se procedează la fel ş.a.m.d.

Deoarece $m \leqslant n$, va exista un m astfel încît o soluţie corespunzătoare să aparţină domeniului Ω. Aceasta va fi tocmai soluţia problemei.

Dacă $\operatorname{rang} \| \alpha_{ik} \| < m$, atunci se calculează soluţia optimă pentru fiecare varietate n-r-dimensională definită de

$$g_j(\mathbf{x}) = 0, \qquad (j = 1,2, \ldots, m).$$

La fel se procedează şi în cazul în care $m > n$, cu o singură modificare privind criteriul de optim pe o muchie.

Scanned with OKEN Scanner

Se observă în primul rînd că algoritmul prezentat este totdeauna finit. În al doilea rînd se vede că acest algoritm prezintă avantaje atunci cînd numărul restricţiilor nu este prea mare.

Formula (22.4) se simplifică mult în cazul cînd funcţia de scop este o sumă de pătrate, adică

$$a_{ik} = \begin{cases} 1 & , \quad i = k \\ 0 & , \quad i \neq k. \end{cases}$$

În acest caz formula (22.4) capătă forma

$$(22.5) \quad x_k = \frac{\sum^{(k)} c_{\nu_1} \dots c_{\nu_{n-m}} \Delta(k_1, \dots, k_m)\, \Delta^{(k)}(k_1, \dots, k_m)}{\sum c_{\nu_1} \dots c_{\nu_{n-m}} [\Delta(k_1, k_2, \dots, k_m)]^2}, \quad (k = 1, 2, \dots, n),$$

unde

$$\Delta(k_1, k_2, \dots, k_m) = \begin{vmatrix} \alpha_{1k_1} \alpha_{1k_2} & \dots & \alpha_{1k_m} \\ \alpha_{2k_1} \alpha_{2k_2} & \dots & \alpha_{2k_m} \\ \dots & \dots & \dots \\ \alpha_{mk_1} \alpha_{mk_2} & \dots & \alpha_{mk_m} \end{vmatrix},$$

iar $\Delta^{(j)}(k_1, k_2, \dots, k_m)$ se obţine din $\Delta(k_1, k_2, \dots, k_m)$ înlocuind α_{ik_j} cu α_i, $i = 1, 2, \dots, m$. Indicii $k_1, k_2, \dots, k_m$ sînt complementari indicilor $\nu_1, \nu_2, \dots, \nu_{n-m}$ adică dacă punem $N = \{1, 2, \dots, n\}$ şi dacă $\nu_1, \dots, \nu_{n-m} \in N_1$ atunci $k_1, k_2, \dots, k_m \in N - N_1$.

Exemplul 22.1. *Să se minimizeze funcţia*

$$f(\mathbf{x}) = x_1^2 + x_2^2 + 2x_3^2$$

supusă restricţiilor

$$g_1(\mathbf{x}) = -x_1 - x_2 - x_3 + 2 \geqslant 0$$
$$g_2(\mathbf{x}) = 2x_1 + x_2 + 3x_3 - 5 \geqslant 0$$
$$g_3(\mathbf{x}) = x_1 - 2x_2 + 2x_3 - 3 \geqslant 0$$

1) Se calculează soluţia corespunzătoare restricţiei $g_1(\mathbf{x}) = 0$ după formula (22.5). Deoarece în cazul nostru

$$c_1 = c_2 = 1, \; c_3 = 2 \text{ şi}$$

Scanned with OKEN Scanner

$$\sum c_{\nu_1} c_{\nu_2} |\alpha_{ik_1}|^2 = c_1c_2\alpha_{13}^2 + c_1c_3\alpha_{11}^2 + c_2c_3\alpha_{12} = 1 + 2 \cdot 1 + 2 \cdot 1 = 5$$

$$\Sigma^{(1)} c_{\nu_1} c_{\nu_2} \alpha_{1k_1}\alpha_1 = c_2c_3\alpha_{11}\alpha_1 = 2(-1)(-2) = 4$$

$$\Sigma^{(2)} c_{\nu_1} c_{\nu_2} \alpha_{1k_1}\alpha_1 = c_1c_3\alpha_{12}\alpha_1 = 2(-1)(-2) = 4$$

$$\Sigma^{(3)} c_{\nu_1} c_{\nu_2} \alpha_{1k_1}\alpha_1 = c_1c_2\alpha_{13}\alpha_1 = 1(-1)(-2) = 2,$$

rezultă că $\mathbf{x}^1 = (4/5,\ 4/5,\ 2/5)$ şi $\mathbf{x}^1 \notin \Omega$, căci $g_2(\mathbf{x}^1) = -7/5$.

Calculînd în acelaşi mod soluţia corespunzătoare restricţiilor $g_2(\mathbf{x}) = 0$, $g_3(\mathbf{x}) = 0$ se găseşte respectiv $\mathbf{x}^2 = (20/19,\ 10/19,\ 15/19)$ şi $\mathbf{x}^3 = (3/7,\ -6/7,\ 3/7)$, care nu aparţin domeniului Ω, deoarece $g_1(\mathbf{x}^2) = -7/19 < 0$, $g_2(\mathbf{x}^3) = -26/7 < 0$.

2) Calculăm acum soluţia corespunzătoare lui $m = 2$, considerînd restricţiile $g_1(\mathbf{x}) = 0$, $g_2(\mathbf{x}) = 0$.

Avem

$$\sum c_{\nu_1} \begin{vmatrix} \alpha_{1k_1} & \alpha_{1k_2} \\ \alpha_{2k_1} & \alpha_{2k_2} \end{vmatrix}^2 = c_1 \begin{vmatrix} \alpha_{12} & \alpha_{13} \\ \alpha_{22} & \alpha_{23} \end{vmatrix}^2 + c_2 \begin{vmatrix} \alpha_{11} & \alpha_{13} \\ \alpha_{21} & \alpha_{23} \end{vmatrix}^2 + c_3 \begin{vmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{vmatrix}^2 =$$
$$= 4 + 1 + 2 = 7,$$

$$\Sigma^{(1)} c_{\nu_1} \begin{vmatrix} \alpha_{1k_1} & \alpha_{1k_2} \\ \alpha_{2k_1} & \alpha_{2k_2} \end{vmatrix} \cdot \begin{vmatrix} \alpha_1 & \alpha_{1k_2} \\ \alpha_2 & \alpha_{2k_2} \end{vmatrix} = \begin{vmatrix} -1 & -1 \\ 2 & 3 \end{vmatrix} \begin{vmatrix} -2 & -1 \\ 5 & 3 \end{vmatrix} + 2 \begin{vmatrix} -1 & -1 \\ 2 & 1 \end{vmatrix} \cdot \begin{vmatrix} -2 & -1 \\ 5 & 1 \end{vmatrix} =$$
$$= 1 + 6 = 7$$

La fel se calculează $\Sigma^{(2)}$ şi $\Sigma^{(3)}$ şi avem

$$\Sigma^{(2)} = \begin{vmatrix} -1 & -1 \\ 1 & 3 \end{vmatrix} \cdot \begin{vmatrix} -2 & -1 \\ 5 & 3 \end{vmatrix} + 2 \begin{vmatrix} -1 & -1 \\ 2 & 1 \end{vmatrix} \cdot \begin{vmatrix} -1 & -2 \\ 2 & 5 \end{vmatrix} = 2 - 2 = 0;$$

$$\Sigma^{(3)} = \begin{vmatrix} -1 & -1 \\ 1 & 3 \end{vmatrix} \cdot \begin{vmatrix} -1 & -2 \\ 1 & 5 \end{vmatrix} + \begin{vmatrix} -1 & -1 \\ 2 & 3 \end{vmatrix} \cdot \begin{vmatrix} -1 & -2 \\ 2 & 5 \end{vmatrix} = 6 + 1 = 7.$$

Prin urmare $\mathbf{x}^4 = (1,\ 0,\ 1)$ şi se vede imediat că $\mathbf{x}^4 \in \Omega$. Deci soluţia optimă este $\mathbf{x}^4$, iar

$$\min_{\mathbf{x} \in \Omega} f(\mathbf{x}) = f(\mathbf{x}^4) = 3$$

Scanned with OKEN Scanner

O aplicație în teoria erorilor

Vom da acum o aplicație a acestei metode în teoria erorilor. Fie dată o funcție reală de n variabile:

$$u = f(x_1, x_2, \ldots, x_n)$$

care admite derivate parțiale de ordinul întîi continue într-o vecinătate a punctului $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)$. Se calculează valoarea

$$u^* = f(x_1^*, x_2^*, \ldots, x_n^*),$$

unde $x_1^*, x_2^* \ldots, x_n^*$ sînt valori aproximative ale argumentelor $x_1, x_2, \ldots x_n$. Una din problemele generale ale teoriei erorilor este următoarea: fiind dat un $\varepsilon > 0$, să aflăm cît de mari pot fi luate abaterile variabilelor independente $\varepsilon_i = x_i - x_i^*$, $i = 1, 2, \ldots, n$, astfel încît

$$|u^* - u| \leqslant \varepsilon. \tag{22.6}$$

Prin urmare, se cere

$$\max \min_{(i)} |\varepsilon_i| \tag{22.7}$$

în condițiile (22.6), care este o problemă de programare matematică. În loc de (22.7) se poate cere

$$\max \sum_{i=1}^{n} \varepsilon_i^2.$$

Vom vedea că (22.6) — (22.7) se poate reduce în anumite condiții la o problemă de programare pătratică de tipul nostru.

Într-adevăr, aplicînd formula creșterilor finite diferenței $u - u^*$, avem

$$u - u^* = \sum_{i=1}^{n} f'_{x_i}(\xi_1, \xi_2, \ldots, \xi_n)\varepsilon_i.$$

În general punctul $(\xi_1, \xi_2, \ldots, \xi_n)$ nu este cunoscut. Se pot însă stabili anumite margini ale derivatelor parțiale. Dacă se ia apoi pentru $f'_{x_i} \approx A_i$,,

Scanned with OKEN Scanner

unde A_i sînt niște numere date, cuprinse între cele două margini, atunci problema noastră se poate enunța astfel: să se găsească

$$\max \sum_{i=1}^{n} \varepsilon_i^2$$

în condițiile

$$\left| \sum_{i=1}^{n} A_i \varepsilon_i \right| \leqslant \varepsilon.$$

Prin urmare, avem o problemă de programare pătratică cu restricții liniare. Această problemă de maxim se poate reduce la o problemă de minim pentru suma

$$\sum_{i=1}^{n} (M - \varepsilon_i)^2$$

unde M este un număr pozitiv destul de mare. Punînd

$$M - \varepsilon_i = X_i,$$

avem problema de programare pătratică:

$$\min \sum_{i=1}^{n} X_i^2$$

în condițiile

$$L_1(\mathbf{X}) = \sum_{i=1}^{n} A_i X_i + \varepsilon - A \geqslant 0,$$

$$L_2(\mathbf{X}) = \sum_{i=1}^{n} (-A_i) X_i + \varepsilon + A \geqslant 0,$$

unde

$$A = M \sum_{i=1}^{n} A_i.$$

Se vede că rangul matricei

$$\begin{pmatrix} A_1 & A_2 & \dots & A_n \\ -A_1 & -A_2 & \dots & -A_n \end{pmatrix}$$

Scanned with OKEN Scanner

este egal cu 1, deci trebuie rezolvate două probleme de optim, corespunzătoare celor două restricţii şi apoi comparate soluţiile.

Luînd prima restricţie $L_1(\mathbf{X}) = 0$, avem

$$\sum c_{\nu_1} \dots c_{\nu_{n-1}} A_i^2 = \sum_{i=1}^{n} A_i^2,$$

$$\sum^{(k)} c_{\nu_1} \dots c_{\nu_{n-1}} A_i(A - \varepsilon) = A_k(A - \varepsilon).$$

Deci

$$X_k^{(1)} = \frac{(A - \varepsilon)A_k}{\sum_{i=1}^{n} A_i^2}, \qquad (k = 1,2 \dots,n). \tag{22.8}$$

În mod analog se calculează soluţia corespunzătoare restricţiei $L_2(\mathbf{X}) = 0$:

$$X_k^{(2)} = \frac{(A + \varepsilon)A_k}{\sum_{i=1}^{n} A_i^2}. \tag{22.9}$$

Întrucît

$$\sum_{k=1}^{n} [X_k^{(2)}]^2 > \sum_{k=1}^{n} [X_k^{(1)}]^2,$$

rezultă că soluţia optimă este cea din (22.8). Prin urmare, abaterile maxime ε_i ale variabilelor independente sînt date de formulele:

$$\varepsilon_k = M - \frac{(A - \varepsilon)A_k}{\sum_{i=1}^{n} A_i^2}.$$

§ 23. Optimizarea funcţiilor pătratice nedefinite

În acest paragraf vom prezenta unele rezultate privitoare la găsirea punctelor de maxim sau minim local şi apoi a celui de maxim respectiv minim global a problemei de programare pătratică de forma:

$$f(\mathbf{x}) = \mathbf{x}'C\mathbf{x} + \mathbf{c}\mathbf{x} + \alpha = \max \ (\min) \tag{23.1}$$

Scanned with OKEN Scanner

în condiţiile

$$\mathbf{Ax} = \mathbf{b}, \qquad \mathbf{x} \geqslant 0, \tag{23.2}$$

unde $\mathbf{C}$ este o matrice pătratică oarecare, simetrică, de ordinul n.

De această problemă s-au ocupat mai mulţi autori. De exemplu ORDEN, A. [65], dă condiţii necesare şi suficiente de existenţă şi unicitate a minimului sau a punctelor staţionare pentru o problemă de programare pătratică nedefinită. Autorul amintit dă şi un algoritm pentru determinarea punctelor de minim local. Mai tîrziu KÜNZI, H.P., KALL, P. şi OETTLI, W. [43] au arătat că condiţiile lui Order sînt necesare şi suficiente de maxim pentru acele soluţii admisibile care sînt în interiorul octantului pozitiv. Pentru punctele frontieră ei şterg componentele lui $\mathbf{A}$, $\mathbf{C}$ şi $\mathbf{c}$ care corespund variabilelor nule şi rezolvă problema nouă obţinută. Dificultatea constă în faptul că aceste variabile care se anulează în punctul staţionar nu sînt cunoscute apriori. De asemenea, metoda nu se aplică cînd nu există minim finit. De remarcat că metoda este greoaie pentru punctele frontieră.

O altă metodă, după părerea noastră mult mai eficace, a fost dată de CANDLER, W. ŞI TOWNSLEY R. J. [9]. Ei dau un algoritm pentru determinarea unui minim local a problemei (23.1)—(23.2), apoi cu ajutorul unui procedeu interesant, bazat pe metoda planelor secante, se micşorează domeniul soluţiilor şi se găseşte un nou minim local. Determinarea unui minim local se face cu ajutorul metodei simplex.

Întrucît metoda ni se pare eficace o vom descrie în cele ce urmează.

Astfel să considerăm problema (23.1)—(23.2) în care presupunem. fără a restrînge generalitatea, că $m < n$ şi coloanele matricei $\mathbf{A}$ sînt liniar independente.

Coloanele matricei $\mathbf{A}$ pot fi privite ca vectori de tipul $(m, 1)$ ce reprezintă cîte un punct în spaţiul euclidian $\mathbf{R}^m$.

1. Calculul unui maxim local

Să presupunem că se dă o soluţie admisibilă $\mathbf{x} \in \Omega$, unde

$$\Omega = \{\mathbf{x} \in \mathbf{R}^n \mid \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}\},$$

care are exact m componente pozitive. Aceasta face posibilă descompunerea vectorului $\mathbf{x}$ sub forma

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_+ \\ \mathbf{x}_0 \end{pmatrix}$$

Scanned with OKEN Scanner

unde $\mathbf{x}_0$ este vectorul nul cu $n - m$ componente, iar $\mathbf{x}_+$ este un vector pozitiv cu m componente. În raport cu această descompunere avem şi

$$\mathbf{A} = (\mathbf{A}_+, \mathbf{A}_0)$$

$$\mathbf{C} = \begin{pmatrix} \mathbf{C}_{++} & \mathbf{C}_{+0} \\ \mathbf{C}_{0+} & \mathbf{C}_{00} \end{pmatrix} \tag{23.3}$$

$$\mathbf{c} = (\mathbf{c}_+, \mathbf{c}_0).$$

Componentele vectorului $\mathbf{x}_+$ se vor numi variabile bazice, iar componentele lui $\mathbf{x}_0$ — variabile nebazice.

Avem deci

$$\mathbf{A}_+\mathbf{x}_+ = \mathbf{b} \tag{23.4}$$

$$f = \mathbf{x}'_+\mathbf{C}_{++}\mathbf{x}_+ + \mathbf{c}_+\mathbf{x}_+ + \alpha. \tag{23.5}$$

Întrucît am presupus că coloanele matricei $\mathbf{A}$ sînt liniar independente, $\mathbf{A}_+$ este inversibilă şi astfel avem

$$\mathbf{x}_+ + \mathbf{A}_+^{-1}\mathbf{A}_0\mathbf{x}_0 = \mathbf{A}_+^{-1}\mathbf{b}, \tag{23.6}$$

de unde

$$\mathbf{x}_+ = \mathbf{A}_+^{-1}\mathbf{b} - \mathbf{A}_+^{-1}\mathbf{A}_0\mathbf{x}_0. \tag{23.7}$$

Dacă $\mathbf{x}'_0 = (0, \ldots, 0, x_{0j}, 0, \ldots, 0)$ atunci

$$\mathbf{x}_+ = \mathbf{A}_+^{-1}\mathbf{b} - \mathbf{A}_+^{-1}\mathbf{a}_{0j}x_{0j} \tag{23.8}$$

unde $\mathbf{a}_{0j}$ este vectorul coloană a matricei $\mathbf{A}_0$ corespunzător variabilei x_{0j}. Din (23.8) se vede că componenta x_{+i} a vectorului $\mathbf{x}_+$ este dată de egalitatea

$$x_{+i} = \mathbf{a}_{+i}\mathbf{b} - \mathbf{a}_{+i}\,\mathbf{a}_{0j}x_{0j}, \tag{23.9}$$

unde $\mathbf{a}_{+i}$ este linia lui $\mathbf{A}_+^{-1}$ corespunzătoare lui x_{+i}.

Valoarea maximă θ_{0j} a lui x_{0j} pentru care $\mathbf{x}_+$ rămîne nenegativ este:

$$\theta_{0j} = \min_{(i)} \{\mathbf{a}_{+i}\mathbf{b}/\mathbf{a}_{+i}\mathbf{a}_{0j}\} \tag{23.10}$$

unde $\mathbf{a}_{+i}\mathbf{a}_{0j} > \mathbf{0}$.

Scanned with OKEN Scanner

Cu ajutorul lui θ_{0j} putem construi acum o nouă soluție admisibilă bazică, care are exact m componente nule. Pentru aceasta e suficient să adăugăm la vectorul $\mathbf{x}$ componenta $\lambda_{0j} = \theta_{0j}$ și să ștergem componenta x_{+i} pentru care s-a atins minimul (23.10) Dacă $\mathbf{a}_{+i}\mathbf{a}_{0j} \leqslant 0$ pentru orice i, atunci domeniul Ω este nemărginit în raport cu x_{0j}.

Să scriem acum funcția de scop (23.5) în funcție de noua soluție admisibilă. Avem

$$(23.11)\quad f = \mathbf{c}_+\mathbf{x}_+ + \mathbf{c}_0\mathbf{x}_0 + \mathbf{x}'_+\mathbf{C}_{++}\mathbf{x}_+ + \mathbf{x}'_+\mathbf{C}_{+0}\,\mathbf{x}_0 + \mathbf{x}'_0\mathbf{C}_{0+}\,\mathbf{x}_+ + \\ + \mathbf{x}_0\,\mathbf{C}_{00}\mathbf{x}_0 + \alpha.$$

Exprimînd variabilele bazice în funcție de cele nebazice din (23.8), obținem

$$(23.12)\qquad f = \mathbf{d}\mathbf{x}_0 + \mathbf{x}'_0\mathbf{B}\mathbf{x}_0 + \beta,$$

unde β este o constantă, $\mathbf{d}$ este un vector de tipul $(1, n-m)$, iar $\mathbf{B} = (b_{ik})$ este o matrice simetrică de tipul $(n-m, n-m)$.

Dacă $\mathbf{x}'_0 = (0,\ldots,0, x_{0j}, 0,\ldots,0)$, atunci din (23.12) obținem:

$$(23.13)\qquad f_{0j} = d_j x_{0j} + b_{jj}x_{0j}^2 + \beta,$$

unde d_j și b_{jj} sînt elementele corespunzătoare lui x_{0j}.

Dacă $b_{jj} < 0 < d_j$, atunci maximul lui f_{0j} din (23.13) este atins cînd x_{0j} ia valoarea

$$(23.14)\qquad x_{0j} = \pi_{0j} = -\,d_j/(2b_{jj}).$$

Dacă $b_{jj} > 0 > d_j$, atunci f_{0j} își atinge în π_{0j} valoarea minimă.

Valoarea maximă a funcției f_{0j} pentru x_{0j} admisibil, adică $0 \leqslant x_{0j} \leqslant \theta_0$, depinde de semnul și mărimea lui b_j și b_{jj}.

Dacă pentru o anumită componentă x_{0j} a lui $\mathbf{x}_0$ valoarea maximă a funcției f_{0j} este atinsă pentru $x_{0j} = \theta_{0j}$, atunci, adăugînd x_{0j} la vectorul $\mathbf{x}_+$ și ștergînd x_{+i} (x_{+i} devine zero cînd $x_{0j} = \theta_{0j}$) obținem o nouă soluție bazică în care funcția de scop este strict mai mare.

Dacă există mai multe componente x_{0j} cu această proprietate, atunci este indicat să se considere aceea care asigură valoarea maximă a funcției de scop.

Procedînd așa vom putea continua procesul algoritmic, majorînd funcția de scop, pînă cînd întîlnim următoarele situații:

1°. *Sau* găsim o soluție $\mathbf{x}_0$ pentru care x_{0j} are proprietatea că $\mathbf{a}_{+i}\mathbf{a}_{0j} \leqslant 0$ pentru orice i și $b_{jj} > 0$, sau $b_{jj} = 0$, $d_j > 0$, în care caz Ω este nemărginit și problema (23.1)—(23.2) n-are soluție, *sau* pentru orice x_{0j} maximul lui f_{0j} este atins pentru $x_{0j} = 0$ sau $x_{0j} = \pi_{0j}$.

Scanned with OKEN Scanner

În ultimul caz se va introduce în baza $\mathbf{x}_+$ componenta lui $\mathbf{x}_0$ pentru care $x_{0j} = \pi_{0j}$. Acest lucru va conduce însă la calcule multe şi ineficiente prin care se vor obţine aproximaţii succesive în jurul punctului de maxim local situat pe o muchie a lui Ω. De aceea ne îndreptăm mai întîi atenţia asupra acelor soluţii nebazice pentru care $b_j > 0$. Pentru aceasta descompunem vectorul $\mathbf{x}_0$ în două componente: $\mathbf{x}_{0*}$ şi $\mathbf{x}_{0\pi}$, unde $\mathbf{x}_{0\pi}$ conţine componentele lui $\mathbf{x}_0$ pentru care maximul lui f_{0j} este atins în $x_{0j} = \pi_{0j}$ iar $\mathbf{x}_{0*}$ restul componentelor. Descompunînd în mod analog vectorul $\mathbf{d}$ şi matricea $\mathbf{B}$, funcţia de scop se va putea scrie

$$f = \mathbf{d}_\pi \mathbf{x}_{0\pi} + \mathbf{x}'_{0\pi} \mathbf{B}_{\pi\pi} \mathbf{x}_{0\pi} + \beta, \tag{23.15}$$

unde s-a pus $\mathbf{x}_{0*} = \mathbf{0}$.

Din modul cum s-a construit vectorul $\mathbf{x}_{0\pi}$ ţinînd seamă de (23.13), rezultă că $\mathbf{B}_{\pi\pi} \leqslant \mathbf{0}$, $\mathbf{d}_\pi > \mathbf{0}$

Punctele staţionare ale lui f din (23.15) sînt date de

$$\mathbf{x}_{0\pi} = \frac{1}{2} \mathbf{B}_{\pi\pi}^{-1} \mathbf{d}'_\pi. \tag{23.16}$$

Dacă anumite componente în (23.16) sînt negative, atunci componentele corespunzătoare se pot şterge în (23.15) şi se rezolvă din nou problema de extremum local pînă cînd se ajunge ca în (23.16) să avem $\mathbf{x}_{0\pi} \geqslant 0$. Vom nota o asemenea soluţie a sistemului (23.16) cu $\mathbf{x}^*_{0\pi}$.

Această soluţie defineşte punctele staţionare ale funcţiei din (23.15), ale cărei variabile sînt supuse restricţiei de nenegativitate.

Descompunînd acum şi matricea $\mathbf{A}_0$ în mod corespunzător în $\mathbf{A}_{0*}$ şi $\mathbf{A}_{0\pi}$, unde $\mathbf{A}_{0\pi}$ corespunde variabilelor din (23.16) nenegative, definim vectorul

$$\mathbf{a}'_\pi = \mathbf{A}_{0\pi} \mathbf{x}^*_{0\pi}. \tag{23.17}$$

Acum se consideră problema iniţială sub următoarea formă extinsă:

$$f = \mathbf{c}^* \mathbf{x}^* + \mathbf{x}^{*\prime} \mathbf{C}^* \mathbf{x}^* + \alpha^* \to \max \tag{23.18}$$

în condiţiile

$$\mathbf{A}^* \mathbf{x}^* = \mathbf{b}^*, \ \mathbf{x}^* \geqslant \mathbf{0}, \tag{23.19}$$

unde

$$\mathbf{x}^* = \begin{pmatrix} \mathbf{x} \\ x_\pi \\ x_\pi^{-1} \end{pmatrix} \qquad \text{de tipul } (n+2,\ 1);$$

Scanned with OKEN Scanner

$$\mathbf{A}^* = \begin{pmatrix} \mathbf{A} & \mathbf{A}_{0\pi}\,\mathbf{x}^*_{0\pi} & \mathbf{0} \\ \mathbf{0} & 1 & \mathbf{0} \end{pmatrix} \qquad \text{de tipul } (m+1,\ n+2);$$

$$\mathbf{b}^* = \begin{pmatrix} \mathbf{b} \\ 1 \end{pmatrix} \qquad \text{de tipul } (m+1,\ 1);$$

$$\mathbf{c}^* = (\mathbf{c},\ \mathbf{c}_{0\pi}\,\mathbf{x}^*_{0\pi},\ 0) \text{ de tipul } (1,\ n+2)$$

şi

$$\begin{pmatrix} \mathbf{C} & \mathbf{C}\mathbf{x}^*_{0\pi} & \mathbf{0} \\ \mathbf{x}^{*\prime}_{0\pi}\,\mathbf{C} & \mathbf{x}^{*\prime}_{0\pi}\,\mathbf{C}\mathbf{x}^*_{0\pi} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \qquad \text{de tipul } (n+2,\ n+2).$$

Ţinînd seama de descompunerea lui $\mathbf{x}$, din (23.19), obţinem

$$\mathbf{A}\mathbf{x} + \mathbf{a}_\pi x_\pi = \mathbf{b},$$

$$x_\pi + x_\pi^{-1} = 1.$$

O soluţie a problemei extinse (23.18)—(23.19) va fi de asemenea o soluţie a problemei iniţiale. Problema extinsă va avea o soluţie bazică formată din $\mathbf{x}_+$ la care se adaugă elementul x_π. Prin urmare, putem defini o nouă bază $\mathbf{x}^{*\prime}_+ = (\mathbf{x}_+,\ x_\pi)$. Mai mult, valoarea funcţiei de scop în $\mathbf{x}^*_+$ va fi mai mare decît în $\mathbf{x}_+$.

Repetînd acest procedeu, vom ajunge eventual la o soluţie bazică în care $\mathbf{d}$ din (23.12) verifică condiţia

$$\mathbf{d} \leqslant \mathbf{0}.$$

Dacă în (23.12) $\mathbf{d} < \mathbf{0}$, atunci am ajuns la un maxim local.

Dacă $\mathbf{d} \leqslant \mathbf{0}$, atunci trebuie analizată matricea $\mathbf{B}$. Pentru aceasta este util să se descompună vectorul $\mathbf{d}$ în două componente

$$\mathbf{d} = (\mathbf{d}_0,\ \mathbf{d}_-),$$

unde $\mathbf{d}_0 = \mathbf{0}$, iar $\mathbf{d}_- < \mathbf{0}$. Descompunînd în mod corespunzător $\mathbf{x}_0$ şi $\mathbf{B}$, relaţia (23.12) se scrie sub forma

$$f = \mathbf{d}_0\mathbf{x}_{00} + \mathbf{d}_-\mathbf{x}_{0-1} + \mathbf{x}'_{00}\,\mathbf{B}_{00}\mathbf{x}_{00} + \mathbf{x}'_{00}\,\mathbf{B}_{0-}\,\mathbf{x}_{0-} + \tag{23.20}$$

$$+ \mathbf{x}'_{0-}\,\mathbf{B}_{-0}\,\mathbf{x}_{00} + \mathbf{x}'_{0-}\,\mathbf{B}_{00}\,\mathbf{x}_{0-} + \beta.$$

Scanned with OKEN Scanner

Dacă se dă lui $\mathbf{x}_0 = 0$ o creştere suficient de mică (considerînd vectorul $\mathbf{dx}_0' = \mathbf{x}_0^{\delta\prime}$), atunci din (23.20) obţinem:

$$df = \mathbf{d}\mathbf{x}_0^{\delta} + 2\mathbf{x}_0^{\delta\prime}\,\mathbf{B}\mathbf{x}_0^{\delta},$$

sau

$$df = (\mathbf{d} + 2\mathbf{x}_0^{\delta\prime}\,\mathbf{B})\mathbf{x}_0^{\delta}. \tag{23.21}$$

Condiţia de nenegativitate asupra lui $\mathbf{x}_0$ implică $\mathbf{x}_0^{\delta} \geqslant 0$. Din (23.21) rezultă că $df \leqslant 0$ dacă

$$\mathbf{d} + 2\mathbf{x}_0^{\delta\prime}\,\mathbf{B} \leqslant 0,$$

sau

$$2\mathbf{x}_0^{\delta\prime}\,\mathbf{B} \leqslant -\mathbf{d}. \tag{23.22}$$

Deoarece $\mathbf{x}_0^{\delta}$ este suficient de mic (23.22) are evident loc pentru vectorul $\mathbf{d}_-$. Pentru vectorul $\mathbf{d}_0$ condiţia (23.22) rămîne verificată dacă

$$2(\mathbf{x}_{00}^{\delta\prime}\,\mathbf{B}_{00} + \mathbf{x}_{0-}^{\delta\prime}\,\mathbf{B}_{-0})\,\mathbf{x}_{00}^{\delta} \leqslant -\mathbf{d}_0\,\mathbf{x}_{00}^{\delta} = 0. \tag{23.23}$$

Dacă $\mathbf{B}_{00}$ este definită negativ sau seminegativ, atunci am ajuns la un maxim local. În caz contrar fie $\mathbf{x}_0^{\delta^*} \geqslant \mathbf{0}$ valoarea lui $\mathbf{x}_0$ pentru care $\mathbf{x}_0^{\delta^*\prime}\,\mathbf{B}\mathbf{x}_0^{\delta^*} > 0$. Atunci se obţine o partiţie analoagă a lui $\mathbf{A}$. Fie $\mathbf{A}_0^{\delta}$ matricea corespunzătoare lui $\mathbf{x}_0^{\delta}$. Definim vectorul de tipul $(1, m)$:

$$\mathbf{a}_d = [\mathbf{A}_0^{\delta}\ \mathbf{x}_0^{\delta^*}]'$$

şi se consideră problema analoagă cu (23.18)—(23.19), cu singura deosebire că acum nu se introduc restricţii noi. Introducerea noii variabile în bază va conduce la o valoare mai mare a funcţiei de scop. Să recapitulăm acum procedeul. Astfel pornind de la o soluţie bazică, se schimbă baza astfel ca funcţia de scop să crească, evitînd pe cît posibil trecerea printr-un punct de maxim local relativ de pe o muchie (adică $x_{0j} = \pi_{0j} < \theta_{0j}$). După ce nu se mai poate evita un asemenea punct de maxim relativ, se consideră toate asemenea variabile nebazice şi se rezolvă noua problemă (23.18)—(23.19).

Dacă matricea amintită $\mathbf{B}_{00}$ este negativ definită, atunci am ajuns la un maxim local. În caz contrar punctul respectiv este un punct şa şi se consideră noua problemă corespunzătoare vectorului $\mathbf{a}_d$.

Scanned with OKEN Scanner

Observaţie. Întrucît o funcţie pătratică admite un singur punct staţionar este foarte probabil să ajungem (chiar după un număr finit de paşi) la un maxim local al funcţiei de scop. Totuşi există cazuri cînd se poate produce un fel de degenerare [9]. De altfel nici convergenţa algoritmului nu este demonstrată.

2. Calculul maximului global

Dacă metoda descrisă mai sus pentru determinarea unui punct de maxim local este destul de greoaie, determinarea punctului de maxim global se bazează pe o ideie interesantă, care poate avea multe alte aplicaţii importante. Unele dintre acestea vor fi date în cele ce urmează. Ideia constă în eliminarea din domeniul soluţiilor Ω a punctului de maxim local găsit, împreună cu o parte din domeniul Ω în care funcţia de scop ia valori mai mici decît în punctul de maxim local găsit. Această micşorare a domeniului soluţiilor se face prin introducerea unei noi restricţii liniare la cele iniţiale, prin care se taie din domeniul Ω o parte Ω_1 şi apoi se caută un nou punct de extremum local pe $\Omega - \Omega_1$.

Presupunem deci că punctul (23.4) este un punct de maxim local al funcţiei de scop dată în (23.12). Dacă în (23.12) o singură variabilă nebazică este pozitivă egală cu t, atunci funcţia de scop se scrie

$$f = dt + Bt^2 + \beta, \tag{23.24}$$

unde β, d şi B sînt numere reale date.

Deoarece am presupus că am pornit dintr-un punct de maxim local al lui f, rezultă că $d < 0$. În acest caz trivial $f = \beta$ este maxim global pentru $t \geqslant 0$ numai dacă $B < 0$.

De aceea considerăm problema

$$f = dt + Bt^2 + \beta \to \max \tag{23.25}$$

pentru $t \geqslant 0$, unde

$$d < 0 < B.$$

În acest caz maximul lui f este $f = \beta$ pentru $t \leqslant \left|\frac{d}{B}\right|$ şi $f = dt + Bt^2 + \beta$ pentru $t > \left|\frac{d}{B}\right|$.

De aceea problema (23.25) poate fi înlocuită cu

$$f = \max(\beta,\ \beta + dt + Bt^2), \tag{23.26}$$

Scanned with OKEN Scanner

cu condiția

$$t > \left| \frac{d}{B} \right| \tag{23.27}$$

unde $d < 0 < B$.

Vom extinde acum ideia de mai sus la cazul general. Pentru aceasta considerăm mai întîi subdomeniul lui Ω pe care diferențiala df este negativă, care, ținînd seamă de (23.21) este dată de inegalitatea:

$$df = (\mathbf{d} + 2\mathbf{x}_0^{*\prime}\mathbf{B})\mathbf{x}_0^{\delta} \geqslant 0, \tag{23.28}$$

unde $\mathbf{x}_0^{\delta} \geqslant \mathbf{0}$ este din nou un vector suficient de mic, iar $\mathbf{x}_0^{*} > \mathbf{0}$ este un vector pozitiv, reprezentînd un vector nebazic.

Inegalitatea (23.28) va fi satisfăcută dacă

$$\mathbf{x}_0^{*\prime}\mathbf{B} \leqslant -\frac{1}{2}\mathbf{d}, \quad \mathbf{x}_0^{*} \geqslant \mathbf{0}. \tag{23.29}$$

Deoarece, prin definiție, într-un maxim local $\mathbf{d} \leqslant \mathbf{0}$, putem înlocui $-\frac{1}{2}\mathbf{d}$ cu $\frac{1}{2}\mathbf{d}|$ (înțelegînd prin $|\mathbf{d}|$ vectorul obținut din $\mathbf{d}$ luînd componentele în valoare absolută).

Dacă pentru un anumit punct $\mathbf{x}_0^0$ avem $df = 0$, atunci în punctul $2\mathbf{x}_0^0$ funcția f ia aceeași valoare ca și în $\mathbf{x} = \mathbf{0}$. Astfel se constată că funcția f *are valoarea maximă* $f = \beta$ *în regiunea*

$$\mathbf{x}_0^{*\prime}\mathbf{B} \leqslant |\mathbf{d}|, \qquad \mathbf{x}_0^{*} \geqslant \mathbf{0}. \tag{23.30}$$

Pentru a putea înlocui restricțiile (23.30) cu una singură se consideră mai întîi cazul cînd $\mathbf{x}_0^{*} = (0, 0, \ldots, x_{0j}^{*}, \ldots, 0)$. Atunci pentru

$$x_{0j}^{**} = \min_i \{d_i/b_{ij},\ b_{ij} > 0\}$$

toate restricțiile (23.30) rămîn verificate. Aici b_{ij} reprezintă componenta matricei $\mathbf{B}$ corespunzătoare lui x_{0j}. Dacă $b_{ij} \leqslant 0$ pentru toți i, atunci se poate lua $x_{0j}^{**} = 0$.

Definim acum un vector $\mathbf{a}^{**}$ de tipul $(1, n)$ în felul următor. Componentele lui $\mathbf{a}^{**}$ corespunzătoare lui $\mathbf{x}_+$ precum și cele corespunzătoare lui $x_{0j} = 0$ din $\mathbf{x}_0$ se iau egale cu zero. Restul componentelor se iau egale cu $1/x_{0j}^{**}$.

Atunci noua restricție care înlocuiește sistemul (23.30) este următoarea:

$$\mathbf{a}^{**}\mathbf{x} \leqslant 1, \tag{23.31}$$

Scanned with OKEN Scanner

Adăugînd această restricţie la restricţiile iniţiale ale problemei se obţine o nouă problemă pentru care domeniul soluţiilor este o parte a lui Ω. Cu această problemă se procedează la fel, găsindu-se un nou punct de maxim local.

Procesul algoritmic se termină atunci cînd se ajunge la o problemă pentru care domeniul soluţiilor admisibile este vid.

După terminarea algoritmului, prin simpla comparare a maximelor locale se găseşte cel global.

§ 24. Optimizarea unui produs de funcţii liniare

În acest paragraf ne vom ocupa de următoarea problemă de programare pătratică nedefinită:

$$\max f(\mathbf{x}) = \max(\mathbf{cx} + \alpha)(\mathbf{dx} + \beta) \tag{24.1}$$

supusă restricţiilor

$$\mathbf{Ax} \leqslant \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}, \tag{24.2}$$

unde $\mathbf{c} = (c_1, c_2, \ldots, c_n)$, $\mathbf{d} = (d_1, d_2, \ldots, d_n)$, $\mathbf{x}' = (x_1, x_2, \ldots, x_n)$, α, β sînt constante, iar matricea $\mathbf{A}$ este de tipul (m, n).

Se presupune că $\mathbf{cx} + \alpha \geqslant 0$, $\mathbf{dx} + \beta \geqslant 0$, cînd

$$\mathbf{x} \in \Omega = \{\mathbf{x} \in \mathbf{R}^n | \mathbf{Ax} \leqslant \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}\},$$

care se presupune nevid şi cu puncte interioare.

De această problemă sau ocupat mai mulţi autori, printre care amintim pe KANTI SWARUP [71] şi PROMILA ANAND şi KANTI SWARUP [2].

1. Reducerea la un program convex

Vom prezenta mai întîi o metodă dată de SWARUP, K. [71] care permite transformarea problemei de programare pătratică nedefinită (24.1)—(24.2) într-o problemă de programare convexă.

Pentru aceasta se consideră transformarea

$$\mathbf{y} = t\mathbf{x}$$

unde $t \geqslant 0$ trebuie determinat astfel ca

$$\mathbf{dy} + \beta t = \delta, \tag{24.3}$$

unde δ este un număr pozitiv dat.

Scanned with OKEN Scanner

Înmulţind şi împărţind funcţia de scop f cu t^2, iar sistemul de inegalităţi (24.2) cu t şi ţinînd seamă de (24.3), ajungem la următoarea problemă

$$\max f(\mathbf{y}, t) = \max \frac{\mathbf{cy} + \alpha t}{t^2} \tag{24.4}$$

supusă restricţiilor

$$\begin{aligned} \mathbf{Ay} - \mathbf{b}t &\leqslant \mathbf{0} \\ \mathbf{dy} + \beta t &= \delta \\ \mathbf{y} \geqslant \mathbf{0},\ t &\geqslant 0\ . \end{aligned} \tag{24.5}$$

Se poate verifica uşor că orice pereche $(\mathbf{y}, t)$ care satisface condiţiile (24.5) are componenta a doua $t > 0$.

De asemenea se demonstrează [71] că dacă matricea $\mathbf{B} = (b_{ik})$ este semipozitiv definită, atunci funcţia

$$f(\mathbf{z}) = \frac{\mathbf{z'Bz} + \alpha_1}{\mathbf{dz} + \beta_1}$$

este *convexă*, dacă suma $\mathbf{dz} + \beta_1$ este pozitivă şi $\alpha_1 \geqslant 0$.

Vom demonstra acum următoarea teoremă.

TEOREMA 24.1. *Dacă* $(\mathbf{y}^*, t^*)$ *este o soluţie optimă a problemei* (24.4) — (24.5), *atunci* $\mathbf{y}^*/t^*$ *este o soluţie optimă a problemei* (24.1)—(24.2).

Demonstraţie. Într-adevăr, să presupunem că teorema nu este adevărată adică există o soluţie optimă $\mathbf{x}^* \in \Omega$ a problemei (24.1)—(24.2) astfel încît să avem

$$(\mathbf{ex}^* + \alpha)(\mathbf{dx}^* + \beta) > \left[\mathbf{e}\left(\frac{\mathbf{y}^*}{t^*}\right) + \alpha\right]\left[\mathbf{d}\left(\frac{\mathbf{y}^*}{t^*}\right) + \beta\right]. \tag{24.6}$$

Atunci există $\eta > 0$ astfel încît

$$\mathbf{dx}^* + \beta = \eta\delta$$

şi considerăm

$$\hat{\mathbf{y}} = \eta^{-1}\mathbf{x}^*,\ \hat{t} = \eta^{-1}.$$

Avem

$$\eta^{-1}[\mathbf{d}\,\mathbf{x}^* + \beta] = \mathbf{d}\,\hat{\mathbf{y}} + \beta\hat{t} = \delta$$

Scanned with OKEN Scanner

şi

$$\eta^{-1}\mathbf{A}x^* - \mathbf{b}\eta^{-1} \leqslant \mathbf{0},$$

adică

$$\mathbf{A}\hat{y} - \mathbf{b}\hat{t} \leqslant \mathbf{0}.$$

Prin urmare $(\hat{y}, \hat{t})$ este o soluţie admisibilă a problemei (24.4)—(24.5). Însă pe de o parte avem

$$(24.7) \qquad (\mathbf{c}x^* + \alpha)(\mathbf{d}x^* + \beta) = \frac{\eta^{-2}(\mathbf{c}x^* + \alpha)(\mathbf{d}x^* + \beta)}{\eta^{-2}} = \frac{(\mathbf{c}\hat{y} + \alpha\hat{t})\delta}{\hat{t}^2},$$

iar pe de altă parte

$$(24.8) \qquad \left[\mathbf{c}\left(\frac{y^*}{t^*}\right) + \alpha\right]\left[\mathbf{d}\left(\frac{y^*}{t^*}\right) + \beta\right] = \frac{(\mathbf{c}y^* + \alpha t^*)(\mathbf{d}y^* + \beta t^*)}{t^{*2}} = \frac{(\mathbf{c}y^* + \alpha t^*)\delta}{t^{*2}}.$$

Din (24.7) şi (24.8), ţinînd seamă de (24.6), rezultă

$$\frac{\mathbf{c}\hat{y} + \alpha\hat{t}}{\hat{t}^2} > \frac{\mathbf{c}y^* + \alpha t^*}{t^{*2}}$$

deoarece $\delta > 0$, ceea ce contrazice ipoteza.

Astfel pentru a rezolva problema dată (24.1)—(24.2) este suficient să se rezolve problema (24.4)—(24.5).

Observăm însă că în condiţiile date

$$\max \frac{\mathbf{c}y + \alpha t}{t^2} = \frac{1}{\min \dfrac{t^2}{\mathbf{c}y + \alpha t}}$$

şi de aceea problema (24.4)—(24.5) este echivalentă cu problema

$$\min \frac{t^2}{\mathbf{c}y + \alpha t}$$

în condiţiile (24.5).

Însă pe baza celor amintite, rezultă că funcţia $t^2/(cy + \alpha t)$ este o funcţie convexă şi astfel rezultă că orice problemă (24.1)—(24.2) poate fi redusă la o problemă de programare convexă.

2. O metodă finită pentru optimizarea produsului

La acest punct vom prezenta o metodă finită pentru rezolvarea problemei:

$$\min f(\mathbf{x}) = \min [(\mathbf{cx} + \alpha)(\mathbf{dx} + \beta)] \tag{24.9}$$

cînd

$$\mathbf{Ax} \leqslant \mathbf{b},\ \mathbf{x} \geqslant 0. \tag{24.10}$$

Acest procedeu se bazează pe faptul că funcţia de scop din cazul nostru este cvasi-concavă în Ω din care cauză îşi atinge minimul într-un vîrf.

Definiţia 24.1. O funcţie $f(\mathbf{x})$ este cvasi-concavă pe Ω dacă oricare ar fi $\mathbf{x}', \mathbf{x}'' \in \Omega$, $\mathbf{x}' \neq \mathbf{x}''$ şi oricare ar fi $t \in [0, 1]$,

$$f[t\mathbf{x}' + (1 - t)\mathbf{x}''] \geqslant \min [f(\mathbf{x}'), f(\mathbf{x}'')],$$

O funcţie f diferenţiabilă pe Ω este cvasi-concavă pe Ω, dacă

$$f(\mathbf{x}') \geqslant f(\mathbf{x}'') \Rightarrow \nabla f(\mathbf{x}'')(\mathbf{x}' - \mathbf{x}'') \geqslant 0,$$

oricare ar fi $\mathbf{x}', \mathbf{x}'' \in \Omega$.

Definiţia 24.2. O funcţie f se numeşte explicit cvasi-concavă pe Ω, dacă pentru orice $\mathbf{x}', \mathbf{x}'' \in \Omega$, $\mathbf{x}' \neq \mathbf{x}''$ satisfăcînd condiţia $f(\mathbf{x}') \neq f(\mathbf{x}'')$ rezultă

$$f[t\mathbf{x}' + (1 - t)\mathbf{x}''] > \min [f(\mathbf{x}'), f(\mathbf{x}'')]$$

oricare ar fi $t \in]0, 1[$.

Funcţiile explicit cvasi-concave au două proprietăţi importante [72]:

a) *Minimul unei funcţii explicit cvasi-concave pe Ω este atins într-un vîrf al poliedrului Ω.*

b) *Orice maxim local al unei funcţii explicit cvasi-concave, este şi maxim global al său pe Ω.*

Să demonstrăm acum următoarea teoremă.

Scanned with OKEN Scanner

TEOREMA 24.2. *Funcţia* $f(\mathbf{x}) = (\mathbf{cx} + \alpha)(\mathbf{dx} + \beta)$ *este explicit cvasi-concavă pe domeniul* $\mathbf{cx} + \alpha > 0$, $\mathbf{dx} + \beta > 0$, $\mathbf{x} \geqslant 0$.

Demonstraaţie. Într-adevăr, să presupunem că $\mathbf{x}'$ şi $\mathbf{x}''$ verifică condiţiile teoremei şi că

$$\mathbf{cx}'' + \alpha)(\mathbf{dx}'' + \beta). \tag{24.11}$$

Atunci avem succesiv:

$$f[t\mathbf{x}' + (1-t)\mathbf{x}''] = \{\mathbf{c}[t\mathbf{x}' + (1-t)\mathbf{x}''] + \alpha\}\{\mathbf{d}[t\mathbf{x}' + (1-t)\mathbf{x}''] +$$

$$+ \beta\} = [t(\mathbf{cx}' + \alpha) + (1-t)(\mathbf{cx}'' + \alpha)]\,[t(\mathbf{dx}' + \beta) + (1-t)(\mathbf{dx}'' + \beta)] =$$

$$= t^2(\mathbf{cx}' + \alpha)(\mathbf{dx}' + \beta) + (1-t)^2(\mathbf{cx}'' + \alpha)(\mathbf{dx}'' + \beta) +$$

$$+ t(1-t)\,[(\mathbf{cx}' + \alpha)(\mathbf{dx}'' + \beta) + (\mathbf{cx}'' + \alpha)(\mathbf{dx}' + \beta)] >$$

$$> t^2(\mathbf{cx}'' + \alpha)(\mathbf{dx}'' + \beta) + (1-t)^2(\mathbf{cx}'' + \alpha)(\mathbf{dx}'' + \beta) +$$

$$+ t(1-t)\,[(\mathbf{cx}' + \alpha)(\mathbf{dx}'' + \beta) + (\mathbf{cx}'' + \alpha)(\mathbf{dx}' + \beta)].$$

Deci

$$f[t\mathbf{x}' + (1-t)\mathbf{x}'] > f(\mathbf{x}'') +$$

$$+ t(1-t)\,[(\mathbf{cx}' + \alpha)(\mathbf{dx}'' + \beta) + (\mathbf{cx}'' + \alpha)(\mathbf{dx}' + \beta) - \tag{24.12}$$

$$-2(\mathbf{cx}'' + \alpha)(\mathbf{dx}'' + \beta)] = f(\mathbf{x}'') + t(1-t)\,\varphi(\mathbf{x}', \mathbf{x}'').$$

Însă dacă

1) $$\mathbf{cx}' + \alpha \geqslant \mathbf{cx}'' + \alpha \text{ şi } \mathbf{dx}' + \beta \geqslant \mathbf{dx}'' + \beta$$

atunci

$$\varphi(\mathbf{x}', \mathbf{x}'') = (\mathbf{cx}'' + \alpha)[(\mathbf{dx}' + \beta) - (\mathbf{dx}'' + \beta)] +$$

$$+ (\mathbf{dx}'' + \beta)[(\mathbf{cx}' + \alpha) - (\mathbf{cx}'' + \alpha)] \geqslant 0$$

şi deci din (7.12) rezultă

$$f[t\mathbf{x}' + (1-t)\mathbf{x}''] > f(\mathbf{x}'').$$

Scanned with OKEN Scanner

Dacă

2) $$\mathbf{cx}' + \alpha \geqslant \mathbf{cx}'' + \alpha \text{ şi } \mathbf{dx}' + \beta \leqslant \mathbf{dx}'' + \beta$$

sau

$$\mathbf{cx}' + \alpha \leqslant \mathbf{cx}'' + \alpha \text{ şi } \mathbf{dx}' + \beta \geqslant \mathbf{dx}'' + \beta$$

atunci

$$\varphi(\mathbf{x}', \mathbf{x}'') > (\mathbf{cx}' + \alpha)(\mathbf{dx}'' + \beta) + (\mathbf{cx}'' + \alpha)(\mathbf{dx}' + \beta) -$$
$$-(\mathbf{cx}'' + \alpha)(\mathbf{dx}'' + \beta) - (\mathbf{cx}' + \alpha)(\mathbf{dx}' + \beta) =$$
$$= [(\mathbf{cx}' + \alpha) - (\mathbf{cx}'' + \alpha)][(\mathbf{dx}'' + \beta) - (\mathbf{dx}' + \beta)] \geqslant 0$$

şi din nou rezultă

$$f[t\mathbf{x}' + (1 - t)\mathbf{x}''] > f(\mathbf{x}'').$$

Deoarece funcţia de scop f din (24.9) este deci explicit cvasi-concavă pe $\Omega = \{\mathbf{x} \in \mathbf{R}^n | \mathbf{Ax} \leqslant \mathbf{b}, \mathbf{x} \geqslant 0\}$, ea îşi atinge, pe baza *a*), minimul local într-un vîrf al poliedrului. Acest lucru face posibilă folosirea tehnicii simplex pentru determinarea unui minim local. O asemenea tehnică a fost folosită, de exemplu, de KANTI SWARUP [72] şi PROMILA ANAND şi KANTI SWARUP [2].

Pentru determinarea minimului absolut se foloseşte metoda lui Candler şi Townsley, descrisă în paragraful precedent, cu care se elimină o parte din domeniul soluţiilor admisibile Ω, în care funcţia de scop ia valori mai mari decît în minimul local găsit.

În cele ce urmează vom da un alt procedeu pentru rezolvarea problemei (24.9)—(24.10) bazat pe varianta simplex a lui Zuhoviţki-Avdeeva dat în paragraful 5, capitolul I, care permite să se opereze cu inegalităţi, fără să fie nevoie să se introducă variabile auxiliare de transformare a inegalităţilor în egalităţi.

Vom considera deci următoarea problemă:

(24.9) $$\min f(\mathbf{x}) = \min [(\mathbf{cx} + \alpha)(\mathbf{dx} + \beta)]$$

pentru $$\mathbf{x} \in \Omega = [\mathbf{x} \in \mathbf{R}^n | \mathbf{Ax} \leqslant \mathbf{b}\},$$

unde $$\Omega \neq \emptyset \text{ şi } \mathbf{cx} + \alpha > 0,\ \mathbf{dx} + \beta > \mathbf{0},\ \mathbf{x} \in \Omega.$$

3. Algoritmul pentru determinarea minimului local

Algoritmul constă în două părţi. În prima parte se determină un punct, de minim local, iar în a doua parte, prin micşorarea domeniului soluţiilor Ω prin introducerea unei noi restricţii se determină minimul global al funcţiei

Scanned with OKEN Scanner

de scop. La acest punct vom descrie algoritmul pentru determinarea unui punct de minim local.

Se scrie tabelul simplex sub forma

(24.13)

	$-x_1$	$-x_2$	$\dots$	$-x_n$	1
$y_1 =$	a_{11}	a_{12}	$\dots$	a_{1n}	b_1
$y_2 =$	a_{21}	a_{22}	$\dots$	a_{2n}	b_2
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$y_m =$	a_{m1}	a_{m2}	$\dots$	a_{mn}	b_m
$z_1 =$	$-c_1$	$-c_2$	$\dots$	$-c_n$	0
$z_2 =$	$-d_1$	$-d_2$	$\dots$	$-d_n$	0

unde $\mathbf{A} = (a_{ik})$, deci se vede că aici $y_i \geqslant 0$, $(i = 1, 2, \dots, m)$.

1°. *Eliminarea variabilelor independente.* Se elimină variabilele independente $x_1, x_2, \dots, x_n$ (presupunînd că rang $\mathbf{A} = n$), efectuînd n paşi Jordan (vezi, §4, capitolul I).

După efectuarea a n paşi Jordan modificaţi şi după o eventuală renumerotare, tabelul (24.13) se va transforma în tabelul:

(24.14)

	$-y_1$	$-y_2$	$\dots$	$-y_n$	1
$y_{n+1} =$	$c_{n+1,1}$	$c_{n+1,2}$	$\dots$	$c_{n+1,n}$	e_{n+1}
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$y_m =$	c_{m1}	c_{m2}	$\dots$	c_{mn}	e_m
$z_1 =$	p_1	p_2	$\dots$	p_n	v_1
$z_2 =$	q_1	q_2	$\dots$	q_n	v_2

unde

$$x_k = -\sum_{j=1}^{n} c_{kj} y_j + e_k, \qquad (k = 1, 2, \dots, n) \tag{24.15}$$

nu au mai fost trecuţi în tabel.

Tabelului (24.14) îi corespunde acum următoarea formă canonică a problemei (24.9): *să se minimizeze funcţia*

$$Z = z_1 z_2 = (-\mathbf{py} + v_1)(-\mathbf{qy} + v_2) \tag{24. 9'}$$

Scanned with OKEN Scanner

supusă restricțiilor

$$\mathbf{Cy} \leqslant \mathbf{e},\ \mathbf{y} \geqslant 0, \tag{24.10'}$$

unde $\mathbf{C} = (c_{n+i,k})$ $(i = 1, 2, \ldots, m-n, k = 1, 2, \ldots, n)$.

De aici se vede că dacă restricțiile Ω conțin de la început inegalitățile $\mathbf{x} \geqslant \mathbf{0}$, atunci această etapă a algoritmului cade, nefiind necesară eliminarea variabilelor independente.

2°. *Găsirea unei soluții admisibile (bazice).* Pentru a găsi o soluție de pornire, (o soluție bazică) se procedează după cum urmează:

1) Dacă în tabelul (7.14) $e_{n+1} \geqslant 0, \ldots, e_m \geqslant 0$, atunci soluția admisibilă este

$$y_1 = y_2 = \ldots = y_n = 0, \qquad y_i = e_i, \qquad (i = n+1, \ldots, m)$$

iar x_k se citesc din (7.15).

2) Dacă există $e_r < 0, r \in \{n+1, \ldots, m\}$, atunci:

a) dacă

$$\forall\, i \in \{1, 2, \ldots, n\} \Rightarrow c_{ri} > 0,$$

atunci sistemul (27.10)' este incompatibil (în cazul nostru aceasta nu poate avea loc căci am presupus că $\Omega \neq \emptyset$).

b) Dacă există $c_{is} < 0$, atunci coloana s este coloană pivot și se alege elementul pivot după regula obișnuită, adică c_{js} este elementul pivot dacă

$$0 \leqslant \frac{c_j}{c_{js}} \leqslant \frac{c_i}{c_{is}}, \qquad (i = n+1, \ldots, m).$$

Cu acest element pivot se efectuează un nou pas Jordan. Deoarece am presupus că $\Omega \neq \emptyset$, rezultă că după un număr finit de pași vom ajunge la o soluție admisibilă. (Cazul de degenerare se tratează în mod obișnuit).

3°. *Găsirea unui minim local.* Să presupunem că $e_{n+1} \geqslant 0, \ldots, e_m \geqslant 0$.

Dacă

$$\forall\, k \in \{1, 2, \ldots, n\} \Rightarrow p_k \leqslant 0,\ q_k \leqslant 0,$$

atunci

$$Z_{\min} = v_1 v_2.$$

Scanned with OKEN Scanner

Într-adevăr, în acest caz

$$z_1 = -\sum_{k=1}^{n} p_k y_k + v_1 \geqslant v_1,$$

$$z_2 = -\sum_{k=1}^{n} q_k y_k + v_2 \geqslant v_2$$

oricare ar fi $y_1 \geqslant 0, \ldots, y_n \geqslant 0$.

2) Dacă există $s \in \{1, 2, \ldots, n\}$ astfel încît $p_s > 0$ sau $q_s > 0$. Atunci:

a) Dacă, de exemplu, $p_s > 0$ şi $q_s > 0$, atunci coloana s va fi coloana pivot şi alegîndu-se un element pivot în mod obişnuit, se va executa un pas Iordan, după care se va ajunge într-un vîrf al poliedrului Ω în care funcţia f ia valoarea strict mai mică decît în vîrful precedent.

Într-adevăr, punînd

$$y_s = t > 0, \qquad y_k = 0, \qquad k \neq s,$$

$$Z_1 = z_1 z_2 = (-p_s t + v_1)(-q_s t + v_2) =$$
$$= p_s q_s t^2 - (p_s v_2 + q_s v_1)t + v_1 v_2.$$

Pentru ca funcţia f să descrească pe această muchie pentru $t > 0$ suficient de mic, este necesar ca

$$p_s v_1 + q_s v_1 > 0. \tag{24.16}$$

Deoarece însă $v_1 > 0$, $v_2 > 0$ şi $p_s > 0$, $q_s > 0$, rezultă că în acest caz (24.16) este verificată şi astfel coloana s este totdeauna coloană pivot, în care alegerea elementului pivot se face în mod obişnuit. Efectuînd un nou pas Jordan se va ajunge la un nou vîrf al poliedrului Ω, corespunzător $t_0 > 0$, în care

$$f(t_0) = p_s q_s t_0^2 - (p_s v_2 + q_s v_1)t_0 + v_1 v_2 < v_1 v_2.$$

Într-adevăr, se poate verifica imediat că funcţia

$$f(t) = p_s q_s t^2 - (p_s v_2 + q_s v_1)t + v_1 v_2 \tag{24.17}$$

descreşte monoton pentru t verificînd inegalităţile:

$$0 \leqslant t \leqslant \frac{1}{2}\left[\frac{v_1}{p_s} + \frac{v_2}{q_s}\right] \tag{24.18}$$

Scanned with OKEN Scanner

Astfel, rădăcinile trinomului $f(t) = 0$ sînt

$$t_1 = \frac{v_1}{p_s}, \quad t_2 = \frac{v_2}{q_s}.$$

Însă avem $z_1(t_1) = 0$, $z_2(t_2) = 0$ şi deoarece am presupus că $z_1 > 0$, $z_2 > 0$, rezultă că

$$0 \leqslant t_0 = \min \frac{e_i}{e_{is}} \leqslant \min \left(\frac{v_1}{p_s}, \frac{v_2}{q_s} \right) \leqslant \frac{1}{2} \left[\frac{v_1}{p_s} + \frac{v_2}{q_s} \right]$$

şi astfel (24.18) fiind verificată, rezultă că

$$Z(t_0) < v_1 v_2,$$

dacă $t_0 > 0$.

b) Dacă $p_s > 0$, $q_s < 0$, atunci oricare ar fi $t > 0$, este verificată inegalitatea

$$p_s q_s t - (p_s v_2 + q_s v_1) < 0 \tag{24.19}$$

dacă are loc (24.16). Prin urmare, în acest caz funcţia $f(t)$ descreşte monoton în raport cu t, oricare ar fi $t > 0$. Astfel, efectuînd un nou pas Jordan cu un element pivot ales în mod obişnuit în coloana s, se va ajunge la un nou vîrf în care $Z(t_0) < v_1 v_2$.

c) Dacă $p_s > 0$; $q_s = 0$, atunci (24.16) are loc şi rămîne valabilă concluzia de la punctul b). Astfel are loc

TEOREMA 24.3. *Vîrful* $y_1 = y_2 = \ldots = y_n = 0$ *al poliedrului* Ω *este un punct de minim local al funcţiei* f *dacă şi numai dacă*

$$k \in \{1, 2, \ldots, n\} \Rightarrow p_k v_2 + q_k v_1 \leqslant 0, \tag{24.20}$$

Demonstraţie. Necesitatea. Să presupunem că $y_1 = y_2 = \ldots = y_n = 0$ este un punct de minim local al funcţiei f, atunci pe baza concluziilor de la punctele a)—c), rezultă că (24.20) trebuie să aibă loc, căci altfel ar exista $s \in \{1, 2, \ldots, n\}$, astfel încît să aibă loc (24.16) (contrara inegalităţii (24.20)) şi deci s-ar putea efectua un pas Jordan, ceea ce contrazice ipoteza.

Suficienţa. Dacă inegalităţile (24.20) au loc, atunci rezultă

$$f = z_1 z_2 = \left(-\sum_{i=1}^{n} p_i y_i + v_1 \right) \left(-\sum_{k=1}^{n} q_k y_k + v_2 \right) =$$

$$= \sum_{i,k=1}^{n} p_i q_k y_i y_k - \left(v_1 \sum_{k=1}^{n} q_k y_k + v_2 \sum_{i=1}^{n} p_i y_i \right) + v_1 v_2.$$

Scanned with OKEN Scanner

Deci

$$(24.21) \qquad Z = \sum_{i,k=1}^{n} p_i q_k y_i y_k - \sum_{k=1}^{n} (v_2 p_k + v_1 q_k) y_k + v_1 v_2 \geqslant v_1 v_2,$$

dacă $y_k \geqslant 0$, $k = 1, 2, \ldots, n$, sînt suficient de mici.

Prin urmare $y_1 = y_2 = \ldots = y_n = 0$ este punct de minim local al funcţiei f.

4. Determinarea punctului de minim global.

Fie deci $\mathbf{x}^0$ (soluţia sistemului $y_1 = y_2 = \ldots = y_n = 0$) un punct de minim local al funcţiei f. Atunci, ţinînd seamă de (24.21), avem

$$(24.22) \qquad \frac{\partial f}{\partial y_j} = q_j \sum_{k=1}^{n} p_k y_k + p_j \sum_{i=1}^{n} q_i y_i - (v_2 p_j + v_1 q_j).$$

Deoarece $\mathbf{x}^0$ este un punct de minim local al funcţiei, condiţiile (24.20) sînt îndeplinite şi deci

$$(24.23) \qquad \forall j \in \{1, 2, \ldots, n\} \Rightarrow \delta_j = v_2 p_j + v_1 q_j \leqslant 0$$

şi există $s \in \{1, 2, \ldots, n\}$ astfel ca $\delta_s < 0$, căci altfel $\mathbf{x}^0$ ar fi un minim global al funcţiei Z.

Relaţiile (24.22) se mai pot scrie sub forma

$$(24.24) \qquad \frac{\partial f}{\partial y_j} = \sum_{k=1}^{n}{}^{(j)} (q_j p_k + p_j q_k) y_k + 2 p_j q_j y_j - \delta_j.$$

Vom separa din Ω o submulţime a sa Ω_1 în care funcţia de scop f ia valori mai mari decît în $\mathbf{x}^0$, adică pe care

$$(24.25) \qquad \forall j \in \{1, 2, \ldots, n\} \Rightarrow \frac{\partial f}{\partial y_j} \geqslant 0.$$

Pentru aceasta este suficient ca în (24.24) să luăm

$$y_j \leqslant \frac{\delta_j}{2 p_j q_j} \qquad \text{şi} \quad y_k = 0, \; k \neq j,$$

dacă $p_j q_j < 0$ și $y_k = 0$, $\forall k \in \{1, 2, \ldots, n\}$ în caz contrar, căci pe baza (24.23) $\delta_j \leqslant 0$, $j = 1, 2, \ldots, n$. Însă în punctul $y^* = (y_1^*, \ldots, y_n^*)$, unde

$$y_j^* = \begin{cases} 0 & , p_j q_j \geqslant 0 \\ \dfrac{\delta_j}{4 p_j q_j} & , p_j q_j < 0 \end{cases}$$

și $y_k^* = 0$, $k \neq j$, avem

$$(24.26) \quad f(y^*) = p_j q_j \frac{\delta_j^2}{4 p_j^2 q_j^2} - \frac{\delta_j^2}{2 p_j q_j} + v_1 v_2 = \frac{-\delta_j^2}{4 p_j q_j} + v_1 v_2 > v_1 v_2$$

dacă $p_j q_j < 0$.

Deoarece funcția f este explicit cvasi-concavă, rezultă că

$$\min_{x \in \Omega_1} f(x) = v_1 v_2$$

unde

$$\Omega_1 = \left\{ x \in \Omega \,\middle|\, \sum_{j=1}^{n} \frac{y_j}{y_j^*} \leqslant 1 \right\}$$

(aici se consideră că în suma Σ lipsesc termenii corespunzători lui $y_j^* = 0$).

Astfel funcția f nu poate admite un alt punct de minim local în Ω_1. Prin urmare orice minim local al funcției f se află eventual în domeniul $\Omega - \Omega_1$. Pentru a găsi un alt minim local se rezolvă următoarea problemă de programare: *să se minimizeze funcția*

$$Z = z_1 z_2$$

supusă restricțiilor

$$(24.27) \quad Ax \leqslant b$$

$$\sum_{j=1}^{n} \frac{y_j}{y_j^*} \geqslant 1.$$

Punctul x^0 va fi un punct de minim global dacă sistemul (24.27) este incompatibil (adică $\Omega_1 = \Omega$). Ținînd însă seama de tabelul (24.14), sistemul (24.27) este echivalent cu sistemul:

$$\sum_{j=1}^{n} c_{ij} y_j + e_j \geqslant 0, \quad (i = n+1, \ldots, m)$$

$$(24.28) \quad \sum_{j=1}^{n} \frac{y_j}{y_j^*} \geqslant 1.$$

Scanned with OKEN Scanner

Deci are loc

TEOREMA 24.4. *Dacă sistemul (24.27) este incompatibil atunci punctul* $\mathbf{x}^0$ *este un punct de minim global al funcţiei* f *pe domeniul* Ω.

Întrucît orice minim local al funcţiei f este atins într-un vîrf al poliedrului Ω, rezultă că după un număr finit de paşi se va ajunge la punctul de minim global al funcţiei.

Pentru ilustrare considerăm următorul exemplu.

Exemplul 24.1. Să se minimizeze funcţia

$$f = (7 - x_1 + x_2)(5 - x_2)$$

supusă restricţiilor

$$x_1 + 4x_2 \leqslant 4$$
$$2x_1 + 3x_2 \leqslant 6$$
$$x_1 \geqslant 0,\ x_2 \geqslant 0.$$

Tabelul simplex corespunzător este

	$-x_1$	$-x_2$	1
$y_1 =$	1	4	4
$y_2 =$	$\underline{\vert 2 \vert}$	3	6
$z_1 =$	1	-1	7
$z_2 =$	0	1	5

1°. Întrucît variabilele x_1, x_2 sînt supuse condiţiei de nenegativitate, nu e necesară etapa de eliminare a variabilelor.

2°. Deoarece coloana termenilor liberi este pozitivă, rezultă că o soluţie admisibilă este $x_1 = x_2 = 0$.

3°. Avem

$$v_1 q_1 + v_2 p_1 = 7 \cdot 0 + 5 \cdot 1 = 5 > 0$$

deci coloana întîi este colona pivot. Elementul pivot din această coloană este 2. Efectuînd un pas Jordan cu acest element pivot, se ajunge la tabelul:

(24.29)

	$-y_2$	$-x_2$	1
$y_1 =$	$-1/2$	$5/2$	1
$x_1 =$	$1/2$	$3/2$	3
$z_1 =$	$-1/2$	$-5/2$	4
$z_2 =$	0	1	5

Scanned with OKEN Scanner

Deoarece avem

$$v_1q_1 + v_2p_1 = 4 \cdot 0 + 5 \cdot (-1/2) = -5/2 < 0$$
$$v_1q_2 + v_2p_2 = 4 \cdot 1 + 5 \cdot (-5/2) = -17/2 < 0$$

rezultă că punctul $y_2 = x_2 = 0$ este un punct de minim local. Prin urmare un punct de minim local este

$$x_1 = 3, \quad x_2 = 0.$$

4°. Întrucît $p_1q_1 = 0$, iar $p_2q_2 = -5/2$,

$$\delta_2 = v_2p_2 + v_1q_2 = 5 \cdot (-5/2) + 4 \cdot 1 = -17/2,$$

rezultă că $y_2^* = 0$, $x_2^* = 17/10$. Deci noua restricție va fi:

$$x_2 \geqslant 17/10$$

Introducînd în tabelul simplex (24.29) această restricție, obținem tabelul:

	$-y_2$	$-x_2$	1
$y_1 =$	$-1/2$	$\boxed{5/2}$	1
$x_1 =$	$1/2$	$3/2$	3
$y_3 =$	0	-1	$-17/10$
$z_1 =$	$-1/2$	$-5/2$	4
$z_2 =$	0	1	5

Trecînd la calculul unei soluții admisibile pentru această problemă, după efectuarea a doi pași Jordan cu elementele pivot încadrate, vom avea succesiv tabelele:

	$-y_2$	$-y_1$	1
$x_2 =$	$-1/5$	$2/5$	$2/5$
$x_1 =$	$\boxed{4/5}$	$-3/5$	$12/5$
$y_3 =$	$-1/5$	$2/5$	$-13/10$
$z_1 =$	-1	1	5
$z_2 =$	$1/5$	$-2/5$	$23/5$

Scanned with OKEN Scanner

	$-x_1$	$-y_1$	1
$x_2 =$	1/4	1/4	1
$y_2 =$	5/4	− 3/4	3
$y_3 =$	1/4	1/4	− 7/10
$z_1 =$	5/4	1/4	8
$z_2 =$	− 1/4	− 1/4	4

Din ultimul tabel se vede că sistemul de restricţii este incompatibil, căci $y_3 < 0$, cînd $x_1 \geqslant 0$, $y_1 \geqslant 0$.

Prin urmare, conform Teoremei 24.4, punctul $x_1 = 3$, $x_2 = 0$ este punctul de minim absolut căutat.

Observaţie. Problema de maxim se poate trata cu metode obişnuite, deoarece funcţia de scop, fiind explicit cvasi-concavă, orice punct de maxim local este totodată şi punct de maxim global al său.

Scanned with OKEN Scanner

CAPITOLUL V

PROGRAMAREA SIMULTANĂ A MAI MULTOR FUNCȚII DE SCOP

§ 25. Programarea pătratică a mai multor funcții de scop

În multe probleme de programare pătratică intervin situații cînd se cere minimizarea sau maximizarea simultană a mai multor funcții de scop, supuse anumitor restricții. În cazul cînd atît funcțiile de scop cît și restricțiile sînt liniare, problema a fost rezolvată de către ZUHOVIȚKI, S.I. [83] și REMEZ, E. I., STEINBERG, A. S. [66].

De cazul în care funcțiile de scop sînt pătratice și restricțiile liniare s-a ocupat MARUȘCIAC, I. și RĂDULESCU, M. [54].

Problema se enunță în felul următor: Se consideră s funcții pătratice

$$f_j(\mathbf{x}) = \mathbf{x}'\mathbf{C}^j\mathbf{x} + \mathbf{d}^j\mathbf{x} + e, \qquad (j = 1, 2, \ldots, s) \tag{25.1}$$

unde $\mathbf{C}^j = (c^j_{ik})$, $i, k = 1, 2, \ldots, n$, sînt matrici simetrice pozitiv-definite, iar $\mathbf{d}^j$ vectori linie de tipul $(1, n)$, și restricțiile obișnuite

$$\mathbf{Ax} + \mathbf{b} \geqslant \mathbf{0}, \tag{25.2}$$

unde $\mathbf{A}$ este o matrice de tipul (m, n) iar $\mathbf{b}$ un vector coloană de tipul $(m, 1)$.

Atunci problema este următoarea: *să se găsească un punct* $\mathbf{x}^* \in \Omega$, *astfel încît*

$$F(\mathbf{x}^*) = \max_{(j)} f_j(\mathbf{x}^*) = \min_{\mathbf{x}\in\Omega} \max_{(j)} f_j(\mathbf{x}). \tag{25.3}$$

Algoritmul construit pentru rezolvarea problemei pornește de asemenea de la un punct arbitrar $\mathbf{x}^0 \in \Omega = \{\mathbf{x} \in \mathbf{R}^n | \mathbf{Ax} + \mathbf{b} \geqslant 0\}$. Dacă în punctul $\mathbf{x}^0$ au loc relațiile

$$f_1(\mathbf{x}^0) = \ldots = f_p(\mathbf{x}^0) > f_k(\mathbf{x}^0), \qquad (k > p;\ p > 1), \tag{25.4}$$

se demonstrează [54] următorul criteriu de optim:

Scanned with OKEN Scanner

TEOREMA 25.1. *Dacă există doi indici* i *şi* j $(1 \leqslant i, j \leqslant p)$, *astfel încît pentru* $\lambda > 0$, *avem*

$$\nabla f_i(\mathbf{x}^0) = -\lambda \nabla f_j(\mathbf{x}^0),$$

atunci $\mathbf{x}^0$ *este un punct optimal.*

Prin urmare, dacă printr-un punct din domeniul Ω trec două suprafeţe de nivel $f_j(\mathbf{x}) = \text{const.}$, cu aceeaşi constantă şi dacă în acest punct ele sînt separate de acelaşi hiperplan tangent, atunci punctul respectiv este optimal, căci în orice direcţie ne-am deplasat una din cele două funcţii creşte.

Se dă un criteriu analog în cazul în care punctul $\mathbf{x}^0$ se găseşte pe frontiera domeniului Ω. Astfel, dacă în punctul $\mathbf{x}^0$ avem relaţiile

$$g_1(\mathbf{x}^0) = \ldots = g_q(\mathbf{x}^0) = 0, \; g_j(\mathbf{x}^0) > 0, \quad (j > q; \; q > 1), \tag{25.5}$$

unde

$$g_i(\mathbf{x}) = \mathbf{a}_i\mathbf{x} + b_i$$

şi în plus relaţiile (25.4), atunci are loc următorul criteriu de optim:

TEOREMA 25.2 *Condiţia necesară şi suficientă pentru ca punctul* $\mathbf{x}^0$ *să fie optimal este ca pentru orice număr pozitiv* $\varepsilon > 0$ *sistemul de inegalităţi*

$$\nabla f_j(\mathbf{x}^0)\mathbf{u} \geqslant \varepsilon > 0 \,, \qquad (j = 1, 2, \ldots, p) \tag{25.6}$$

$$\mathbf{a}_i\mathbf{u} \geqslant 0 \,, \qquad (i = 1, \ldots, q) \tag{25.7}$$

să fie incompatibil.

Din punct de vedere geometric incompatibilitatea sistemului (25.6) — — (25.7) arată că în punctul $\mathbf{x}^0$ nu există nici o direcţie admisibilă (pentru care să fie verificat sistemul (25.7) şi pe care să descrească toate funcţiile de scop (să fie verificate inegalităţile (25.6)).

Avînd în vedere cele două criterii de optim, algoritmul se continuă în felul următor: Dacă punctul $\mathbf{x}$ nu este un punct optimal atunci se găseşte o direcţie admisibilă de scufundare, rezolvînd următoarea problemă de programare liniară: *să se maximizeze* funcţia

$$z = \varepsilon$$

în condiţiile (25.6) — (25.7) la care se mai adaugă condiţiile

$$|\mathbf{u}| \leqslant M,$$

unde M *este un număr real pozitiv, suficient de mare.*

Scanned with OKEN Scanner

Dacă $\mathbf{u}^0$ este soluţia acestei probleme, atunci ne deplasăm pe această direcţie pînă ce întîlnim fie un nou plan $g_j(\mathbf{x}) = 0$, fie că întîlnim un punct în care devin egale între ele două funcţii de scop. Adică considerăm punctele de forma

$$\mathbf{x} + t\mathbf{u}^0 \qquad (t > 0)$$

şi fie t_i' valorile parametrului pentru care

$$g_i(\mathbf{x}^0 + t_i'\mathbf{u}^0) = 0, \qquad (i = q+1, \ldots, n)$$

iar t_{ij}'' valorile parametrului t în care

$$f_j(\mathbf{x}^0 + t_{ij}''\mathbf{u}^0) = f_i(\mathbf{x}^0 + t_{ij}''\mathbf{u}^0). \qquad (i, j = 1, 2, \ldots, s)$$

Dacă $\tau' = \min\limits_{(t_i'>0)} t_i'$, $\tau'' = \min\limits_{(t_{ij}''>0)} t_{ij}''$,

iar

$$\tau = \min\,(\tau', \tau''),$$

atunci următorul punct de aproximaţie este

$$\mathbf{x}^1 = \mathbf{x}^0 + \tau\mathbf{u}^0,$$

cu care se procedează apoi la fel.

Se demonstrează [54] că acest algoritm este totdeauna convergent şi conduce la soluţia problemei iniţiale.

§ 26. O problemă de programare liniară generală cu mai multe funcţii de scop

În problemele economice care conduc la maximizarea sau minimizarea simultană a mai multor funcţii de scop, problema eficienţei se poate pune sub diverse forme. Astfel, de exemplu, se poate cere ca cea mai mică dintre funcţiile de scop să fie maximă, atunci se ajunge la problema de tipul celei tratate la punctul precedent.

Acest mod de a pune problema nu are însă totdeauna o justificare economică, deoarece funcţiile economice de scop nu pot fi totdeauna comparabile. Ca de exemplu, cînd se cere să se maximizeze volumul producţiei şi simultan să se minimizeze preţul de cost. Aceste două cantităţi nu pot fi comparate direct. Se pot însă găsi anumiţi indicatori cu ajutorul cărora cele două funcţii pot fi comparate. Această metodă de comparare comportă însă greutăţi practice, din care cauză devine greu de aplicat.

Scanned with OKEN Scanner

În anumite condiţii se poate stabili o relaţie (o funcţie) între funcţiile de scop considerate şi atunci problema se reduce la o problemă obişnuită de programare matematică cu o singură funcţie de scop.

Un mod de a pune problema mult mai natural, a fost formulat pentru prima dată de către CHARNES, A. şi COOPER, W. [10] în anul 1957, şi constă în problema determinării aşa numitelor *puncte eficiente* ale domeniului restricţiilor Ω în raport cu funcţiile de scop considerate. Mai tîrziu, în anul 1963 problema aceasta a fost reluată de matematicianul maghiar BÓD, P. [7] care dă şi un algoritm de determinare a punctelor eficiente în raport cu un sistem dat de funcţii de scop.

Această problemă a fost apoi generalizată pentru spaţii abstracte de KOLUMBAN, I. [38] care dă şi criterii de caracterizare a elementelor eficiente.

Trebuie să menţionăm că această problemă este în strînsă legătură cu problema determinării aşa numitelor *infraelemente*, care la noi au fost destul de mult studiate de autor şi Kolumban I.

Într-o lucrare recentă a lui MARUŞCIAC, I. şi RĂDULESCU, M. [55] se studiază şi se prezintă un nou algoritm pentru găsirea tuturor punctelor eficiente în cazul în care sistemul funcţiilor de scop este liniar.

Astfel se consideră o vector-funcţie,

$$\mathbf{y} = \mathbf{C}\mathbf{x}, \tag{26.1}$$

unde $\mathbf{x}$ şi $\mathbf{y}$ sînt vectori coloană respectiv de tipul $(n, 1)$ şi $(p, 1)$, iar $\mathbf{C}$ este o matrice dată de tipul (p, n). Domeniul restricţiilor se defineşte şi aici prin sistemul de inegalităţi liniare

$$\mathbf{A}\mathbf{x} \leqslant \mathbf{b}, \tag{26.2}$$

unde $\mathbf{A}$ este o matrice de tipul (m, n), iar $\mathbf{b}$ un vector coloană de tipul $(m, 1)$.

Fie

$$Y = \{\mathbf{y} \in \mathbf{R}^n \mid \mathbf{y} = \mathbf{C}\mathbf{x};\ \mathbf{x} \in \Omega\},\ \Omega = \{\mathbf{x} \in \mathbf{R}^p \mid \mathbf{A}\mathbf{x} \leqslant \mathbf{b}\}.$$

Se introduce în mulţimea Y o relaţie de ordine:

$$\mathbf{y}' \leqslant \mathbf{y}'' \text{ dacă } \forall k \in \{1, 2, \ldots, p\} \Rightarrow y'_k \leqslant y''_k.$$

Aici prin inegalitate strictă: $\mathbf{y}' < \mathbf{y}''$ se înţelege că $\mathbf{y}' \leqslant \mathbf{y}''$, însă există $j \in \{1, 2, \ldots, p\}$, astfel încît $y'_j < y''_j$.

Un punct $\mathbf{x}^0 \in \Omega$ se numeşte *eficient în raport cu* Y, dacă pentru orice punct $\mathbf{x} \in \Omega$, comparabil cu $\mathbf{x}^0$, din

$$\mathbf{y} = \mathbf{y}(\mathbf{x}) \geqslant \mathbf{y}^0 = \mathbf{y}(\mathbf{x}^0),$$

Scanned with OKEN Scanner

unde

$$\mathbf{y} = \mathbf{C}\mathbf{x}, \quad \mathbf{y}^0 = \mathbf{C}\mathbf{x}^0,$$

rezultă

$$\mathbf{y} = \mathbf{y}^0.$$

Prin urmare, punctul $\mathbf{x}^0 \in \Omega$ este eficient, dacă nu există nici un punct $\mathbf{x} \in \Omega$ comparabil cu $\mathbf{x}^0$ în care toate funcţiile $y_k(\mathbf{x})$ $(k = 1, 2, \ldots, p)$ să fie cel puţin egale cu $y_k(\mathbf{x}^0)$ $(k = 1, 2, \ldots, p)$, dintre care cel puţin una să fie strict mai mare.

Să considerăm acum problema următoare: *să se găsească un punct* $\mathbf{x}^0 \in \Omega$, *astfel încît*

$$\forall \mathbf{x}, \ \mathbf{x} \in \Omega \Rightarrow f(\mathbf{x}^0) \geqslant f(\mathbf{x}), \tag{26.3}$$

unde

$$f(\mathbf{x}) = \min_{(k)} y_k(\mathbf{x}).$$

Un punct $\mathbf{x}^0 \in \Omega$ se va numi *optimal, în raport cu* Y, *dacă* el verifică condiţia (26.3).

Se arată [55] că orice punct optimal în raport cu Y este în acelaşi timp şi eficient faţă de acelaşi sistem Y. Reciproca însă nu este adevărată. Există puncte eficiente care nu sînt optimale. Se pot da şi alte exemple de probleme ale căror soluţii sînt de asemenea puncte eficiente. Ceea ce arată că problema găsirii punctelor de eficienţă este o problemă foarte generală.

În ce priveşte mulţimea punctelor eficiente, se arată [55] că aceasta este totdeauna o mulţime simplu conexă. Există însă cazuri cînd mulţimea punctelor eficiente faţă de sistemul Y coincide cu întregul domeniu al soluţiilor Ω.

În lucrarea autorilor, sus amintită, se dau mai întîi două teoreme de caracterizare a punctelor eficiente în raport cu Y, după care se elaborează un algoritm totdeauna convergent, care permite determinarea punctelor eficiente.

Un criteriu de eficienţă se poate enunţa în felul următor:

TEOREMA 26.1 *Un punct* $\mathbf{x}^0 \in \Gamma_q$, *unde*

$$\Gamma_q = \{\mathbf{x} | \mathbf{a}_i\mathbf{x} = b_i, \quad i = 1, 2, \ldots, q; \quad \mathbf{a}_j\mathbf{x} < b_j, \ j > q\}$$

Scanned with OKEN Scanner

este eficient în raport cu Y, dacă şi numai dacă sistemul de inegalităţi

$$\mathbf{g}^k\mathbf{u} - \varepsilon_k \geqslant 0, \qquad (k = 1, 2, \ldots, p)$$

$$\mathbf{a}_i\mathbf{u} \leqslant 0, \qquad (i = 1, 2, \ldots, q)$$

este incompatibil, oricare ar fi $\varepsilon_k \geqslant 0$ *cu cel puţin unul pozitiv.* Aici $\mathbf{g}^k = (c_{k1}, c_{k2}, \ldots, c_{kn})$.

Algoritmul propus se aseamănă cu cel prezentat la punctul precedent şi constă în iterarea a doi paşi fundamentali. La primul pas se face verificarea dacă punctul este sau nu eficient, iar la al doilea se găseşte o direcţie admisibilă de scufundare, pe care toate funcţiile sistemului Y cresc.

§ 27. Structura mulţimii punctelor de eficienţă a mai multor funcţii de scop

În acest paragraf vom prezenta cîteva rezultate privitoare la structura mulţimii punctelor de eficienţă a unui program cu mai multe funcţii de scop, conţinute în lucrarea [56].

Am văzut în paragraful precedent că în cazul cînd funcţiile de scop sînt liniare, mulţimea punctelor de eficienţă este o mulţime simplu conexă de pe frontiera domeniului admisibil sau tot domeniul Ω. Prin urmare, în acest caz, dacă un punct din interiorul lui Ω este eficient atunci tot domeniul Ω este format din puncte eficiente. Mai interesant se prezintă cazul cînd funcţiile de scop sînt neliniare, deoarece într-un asemenea caz mulţimea punctelor de eficienţă poate fi o submulţime a interiorului lui Ω şi se poate vorbi de aşa numitele curbe şi suprafeţe eficiente pentru un program dat.

1, Definiţie şi exemple

Astfel, să considerăm vector-funcţia

$$\mathbf{y} = \mathbf{f}(\mathbf{x}), \tag{27.1}$$

unde

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_p \end{pmatrix} \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

Scanned with OKEN Scanner

și $y_j = f_j(\mathbf{x}) = f_j(x_1, x_2, \ldots, x_n)$ $(j = 1, 2, \ldots, p)$ sînt funcții continue și convexe pe domeniul soluțiilor admisibile:

$$\Omega = \{\mathbf{x} \in \mathbf{R}^n \mid \mathbf{A}\mathbf{x} \leqslant \mathbf{b}\}. \tag{27.2}$$

Fie

$$Y = \{\mathbf{y} \in \mathbf{R}^p \mid \mathbf{y} = \mathbf{f}(\mathbf{x}),\ \mathbf{x} \in \Omega\}.$$

În mulțimea Y se introduce aceeași relație de ordine ca în § 26. Vom spune că $\mathbf{x}', \mathbf{x}'' \in \Omega$ sînt comparabile prin $\mathbf{f}$ dacă punctele $\mathbf{y}' = \mathbf{f}(\mathbf{x}')$ și $\mathbf{y}'' = \mathbf{f}(\mathbf{x}'')$ sînt comparabile în Y, adică are loc una din următoarele relații

$$\mathbf{y}' \leqslant \mathbf{y}'',\ \mathbf{y}' \geqslant \mathbf{y}'' \text{ sau } \mathbf{y}' = \mathbf{y}''.$$

DEFINIȚIA 27.1. *Un punct* $\mathbf{x}^0 \in \Omega$ *se numește eficient în raport cu* Y, *dacă nu există nici un punct* $\mathbf{x} \in \Omega$ *comparabil cu* $\mathbf{x}^0$ *prin* $\mathbf{f}$ *astfel încît*

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) < \mathbf{y}^0 = \mathbf{f}(\mathbf{x}^0)\ . \tag{27.3}$$

Prin urmare, un punct $\mathbf{x}^0 \in \Omega$ este eficient în raport cu Y dacă oricare ar fi $\mathbf{x} \in \Omega$ comparabil prin $\mathbf{f}$ cu $\mathbf{x}^0$ avem

$$\forall j \in \{1, 2, \ldots, p\} \Rightarrow y_j(\mathbf{x}) \geqslant y_j(\mathbf{x}^0).$$

Dăm cîteva exemple de puncte eficiente:

a) Fie

$$F(\mathbf{x}) = \max_{(j)} f_j(\mathbf{x}).$$

Un punct $\mathbf{x}^0 \in \Omega$ se numește *punct optimal în raport cu* Y dacă

$$\forall \mathbf{x} \in \Omega \Rightarrow F(\mathbf{x}) \geqslant F(\mathbf{x}^0). \tag{27.4}$$

Se poate verifica ușor că *orice punct optimal în raport cu* Y *este și eficient în raport cu* Y.

Într-adevăr, dacă $\mathbf{x}^0 \in \Omega$ este un punct optimal în raport cu Y și dacă notăm cu $X_0 \subseteq \Omega$ mulțimea punctelor comparabile cu $\mathbf{x}^0$ prin $\mathbf{f}$, atunci ținînd seamă de (27.4) avem în particular

$$\forall \mathbf{x} \in X_0 \Rightarrow F(\mathbf{x}) \geqslant F(\mathbf{x}^0). \tag{27.5}$$

Scanned with OKEN Scanner

Însă deoarece $\mathbf{x} \in X_0$ este comparabil cu $\mathbf{x}^0$ prin $\mathbf{f}$ rezultă că

$$\forall j \in \{1, 2, \ldots, p\} \Rightarrow f_j(\mathbf{x}) \geqslant f_j(\mathbf{x}^0), \; f_j(\mathbf{x}) \leqslant f_j(\mathbf{x}^0) \quad \text{sau}$$
$$f_j(\mathbf{x}) = f_j(\mathbf{x}^0).$$

Din (27.5) avem succesiv:

$$f_j(\mathbf{x}^0) \leqslant f_{j_0}(\mathbf{x}^0) = F(\mathbf{x}^0) \leqslant F(\mathbf{x}) = f_{j_1}(\mathbf{x}).$$

În particular are loc inegalitatea:

$$f_{j_1}(\mathbf{x}^0) \leqslant f_{j_1}(\mathbf{x})$$

ţś deoarece $\mathbf{x}$ este comparabil cu $\mathbf{x}^0$ prin $\mathbf{f}$ rezultă

$$f_j(\mathbf{x}^0) \leqslant f_j(\mathbf{x}), \qquad (j = 1, 2, \ldots, p),$$

adică

$$\forall \mathbf{x} \in X_0 \Rightarrow \mathbf{y} = \mathbf{f}(\mathbf{x}) \geqslant \mathbf{y}_0 = \mathbf{f}(\mathbf{x}^0).$$

Prin urmare $\mathbf{x}^0$ este un punct eficient în raport cu Y.

b) Considerăm o aplicaţie

$$F : \mathbf{R}^p \to \mathbf{R}$$

monoton crescătoare în raport cu relaţia de ordine parţială din Y, adică

$$\forall \mathbf{y}', \mathbf{y}'' \in Y, \; \mathbf{y}' < \mathbf{y}'' \Rightarrow F(\mathbf{y}') < F(\mathbf{y}'').$$

Un punct $\mathbf{x}^0 \in \Omega$ se numeşte *punct extremal al funcţiei F în raport cu Y* dacă

$$\forall \mathbf{x} \in X_0 \Rightarrow F[\mathbf{y}(\mathbf{x})] \geqslant F[\mathbf{y}(\mathbf{x}^0)]. \tag{27.6}$$

Se poate verifica uşor [56] că orice punct extremal al lui F în raport cu Y este şi punct eficient în raport cu Y.

Ca exemplu de funcţie F se poate considera media ponderată de ordin impar:

$$F(\mathbf{y}) = \left[\sum_{j=1}^{p} \mu_j y_j^{2q+1}\right]^{\frac{1}{2q+1}}$$

Scanned with OKEN Scanner

unde q este număr întreg nenegativ iar $\mu_j > 0$ cu $\Sigma\mu_j = 1$, care este monoton crescătoare pe Y.

2. Criterii de eficiență

Fie $\mathbf{x} \in \Omega$. Notăm

$$\mathbf{g}^j(\mathbf{x}) = -\nabla f_j(\mathbf{x}), \qquad (j = 1, 2, \ldots, p)$$

$$\Gamma_q = \{\mathbf{x} \in \Omega | \forall i \in \{1, 2, \ldots, q\} \Rightarrow \mathbf{a}^i\mathbf{x} = b_i, \ \forall j > q \Rightarrow \mathbf{a}^j\mathbf{x} < b_j\},$$

unde $\mathbf{a}^i = (a_{i1}, a_{i2}, \ldots, a_{in})$, iar $q \in \{1, 2, \ldots, m\}$.

TEOREMA 27.1. *Un punct $\mathbf{x}^0 \in \Gamma_q$ este eficient în raport cu Y dacă și numai dacă sistemul de inegalități*

$$\begin{aligned} \mathbf{g}^j(\mathbf{x}^0)\mathbf{u} - \varepsilon_j &\geqslant 0, \qquad (j = 1, 2, \ldots, p) \\ \mathbf{a}^i\mathbf{u} &\leqslant 0, \qquad (i = 1, 2, \ldots, q) \end{aligned} \tag{27.7}$$

este incompatibil oricare ar fi $\varepsilon' = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_p) \neq \mathbf{0}$, $\varepsilon' \geqslant \mathbf{0}$.

Demonstrație. Necesitate. Fie $\mathbf{x}^0 \in \Gamma_q$ un punct eficient în raport cu Y. Atunci oricare ar fi $\mathbf{x} \in X_0$, avem

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) \geqslant \mathbf{y}^0 = \mathbf{f}(\mathbf{x}^0),$$

adică oricare ar fi $\mathbf{x} \in X_0$ în vecinătatea lui $\mathbf{x}^0$ și $\mathbf{u} = [\mathbf{x}^0, \mathbf{x}]$, avem

$$\forall j \in \{1, 2, \ldots, p\} \Rightarrow \mathbf{g}^j(\mathbf{x}^0)\mathbf{u} \leqslant 0. \tag{27.8}$$

Să presupunem acum contrariul, că sistemul (27.7) este compatibil pentru $\varepsilon^0 \geqslant \mathbf{0}$, $\varepsilon^0 \neq \mathbf{0}$.

Fie $\mathbf{u}^0$ o soluție a sistemului (27.7) pentru $\varepsilon = \varepsilon^0$
Deoarece

$$\begin{aligned} \mathbf{g}^j(\mathbf{x}^0)\mathbf{u}^0 &\geqslant \varepsilon_j^0 \geqslant 0, \qquad . (j = 1, 2, \ldots, p), \\ \mathbf{a}^i\mathbf{u}^0 &\leqslant 0, \qquad (i = 1, 2, \ldots, q), \end{aligned} \tag{27.9}$$

rezultă că în direcția $\mathbf{u}^0$ toate funcțiile descresc și cel puțin una, în sens strict. Prin urmare în vecinătatea lui $\mathbf{x}^0$ există un punct $\mathbf{x} \in X_0$ în care are loc (27.9) pentru $\mathbf{u}^0 = k[\mathbf{x}^0, \mathbf{x}]$, $k > 0$, ceea ce contrazice (27.8).

Suficiență. Dacă în punctul $\mathbf{x}^0 \in \Omega$ sistemul (27.7) este incompatibil oricare ar fi vectorul $\varepsilon \geqslant \mathbf{0}$, $\varepsilon \neq \mathbf{0}$, atunci $\mathbf{x}$ fiind un punct arbitrar din

Scanned with OKEN Scanner

X_0, considerăm direcţia $\mathbf{u}^0 = [\mathbf{x}^0, \mathbf{x}]$. Din incompatibilitatea sistemului (27.7) oricare ar fi vectorul pozitiv nenul ε, rezultă că există $j_0 \in \{1, 2, \ldots p\}$ astfel încît

$$\mathbf{g}^{j_0}(\mathbf{x}^0)\mathbf{u} \leqslant 0,$$

adică

$$y_{j_0} = f_{j_0}(\mathbf{x}) \geqslant y^0_{j_0} = f_{j_0}(\mathbf{x}^0).$$

Din această inegalitate, ţinînd seamă de faptul că $\mathbf{x}$ este comparabil prin $\mathbf{f}$ cu $\mathbf{x}^0$, avem

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) \geqslant \mathbf{y}^0 = \mathbf{f}(\mathbf{x}^0),$$

ceea ce arată că $\mathbf{x}^0$ este un punct eficient în raport cu Y.

TEOREMA 27.2. Fie $\mathbf{x}^0 \in \Omega$. *Dacă există* $i, k \in \{1, 2, \ldots, p\}$ *astfel încît*

$$\mathbf{g}^i(\mathbf{x}^0) = -\lambda \mathbf{g}^k(\mathbf{x}^0), \qquad \lambda > 0, \tag{27.10}$$

atunci $\mathbf{x}_0$ *este un punct eficient în raport cu* Y.

Demonstraţie. Într-adevăr, oricare ar fi $\mathbf{u} \in \mathbf{R}^n$ pentru care

$$\mathbf{g}^i(\mathbf{x}^0)\mathbf{u} > \mathbf{0}$$

rezultă

$$\mathbf{g}^k(\mathbf{x}^0)\mathbf{u} < 0,$$

ceea ce arată că sistemul (27.7) nu poate fi compatibil pentru nici un vector $\varepsilon \geqslant \mathbf{0}$, $\varepsilon \neq \mathbf{0}$. Prin urmare, din Teorema 27.1, rezultă că $\mathbf{x}^0$ este un punct eficient în raport cu Y.

Ca o consecinţă a Teoremei 27.2 se pot construi anumite curbe eficiente, adică arce de curbă din Ω formate din puncte eficiente în raport cu Y. Astfel, scriind explicit condiţia (27.10) într-un punct $\mathbf{x} \in \mathbf{R}^n$ avem

$$\frac{\dfrac{\partial f_i}{\partial x_1}}{\dfrac{\partial f_k}{\partial x_1}} = \frac{\dfrac{\partial f_i}{\partial x_2}}{\dfrac{\partial f_k}{\partial x_2}} = \ldots = \frac{\dfrac{\partial f_i}{\partial x_n}}{\dfrac{\partial f_k}{\partial x_n}} = -\lambda, \quad \lambda > 0 \tag{27.11}$$

Scanned with OKEN Scanner

Sistemul (27.11) (dacă este compatibil) permite exprimarea coordonatelor punctului $\mathbf{x} \in \mathbf{R}^n$ în care sînt verificate condiţiile (27.11) în funcţie de parametrul λ sub forma

$$x_j = x_j(\lambda), \qquad (j = 1, 2, \ldots, n).$$

Prin urmare, locul geometric al punctelor din $\mathbf{R}^n$ în care este verificată condiţia (27.10) este un arc de curbă

$$\mathbf{x} = \mathbf{x}(\lambda), \qquad \lambda \in [0, +\infty]. \tag{27.12}$$

Din Teorema 27.2 rezultă că toate punctele arcului (27.12) situate în Ω sînt puncte eficiente în raport cu $\mathbf{Y}$, de aceea, partea arcului

$$C_{ik} = \Gamma_{ik} \cap \Omega,$$

unde Γ_{ik} este arcul (27.12), se numeşte *curbă eficientă a funcţiilor de scop* f_i şi f_k.

Din (27.11) se vede imediat că pentru $\lambda \to 0$ avem

$$\frac{\partial f_i}{\partial x_j} = 0, \qquad (j = 1, 2, \ldots, n),$$

iar pentru $\lambda \to +\infty$.

$$\frac{\partial f_k}{\partial x_j} = 0, \qquad (j = 1, 2, \ldots, n),$$

ceea ce arată că curba de eficienţă C_{ik} este intersecţia cu domeniul soluţiilor admisibile Ω a arcului Γ_{ik} care are extremităţile în punctele de minim absolut al funcţiilor f_i şi f_k.

În general există C_p^2 curbe eficiente, care se obţin intersectînd cu Ω arcele de curbă corespunzătoare cu extremităţile în cele p puncte de extremum absolut al funcţiilor de scop.

Pentru determinarea efectivă a curbei de eficienţă C_{ik} se determină parametrul λ din (27.12) din condiţia

$$\mathbf{A}\mathbf{x}(\lambda) \leqslant \mathbf{b}$$
$$0 \leqslant \lambda \leqslant +\infty.$$

Dacă soluţia acestui sistem de inegalităţi este $\lambda \in [\lambda_1, \lambda_2]$, atunci curba de eficienţă C_{ik} este dată de ecuaţia

$$\mathbf{x} = \mathbf{x}(\lambda), \qquad \lambda \in [\lambda_1, \lambda_2]$$

Scanned with OKEN Scanner

Pentru determinarea tuturor curbelor eficiente se repetă procedeul de mai sus pentru fiecare pereche de funcţii de scop.

Pentru ilustrare considerăm următorul exemplu:

Exemplul 27.1. Să se găsească curbele eficiente în raport cu

$$\mathbf{Y} = \{\mathbf{y} \in \mathbf{R}^2 \mid \mathbf{y} = \mathbf{f}(\mathbf{x}),\ \mathbf{x} \in \Omega\},$$

unde

$$\mathbf{f}(\mathbf{x}) = (f_1(x_1, x_2), f_2(x_1, x_2)),$$

$$f_1(\mathbf{x}) = x^2 + 4x_2^2 - 16$$

$$f_2(x) = 16x_1^2 + 4x_2^2 - 16x_1 - 4x_2 - 59,$$

iar Ω este poligonul definit de sistemul de inegalităţi:

$$\begin{aligned} 2x_1 + 2x_2 - 1 &\geqslant 0 \\ - x_2 + 2 &\geqslant 0 \\ -2x_1 - x_2 + 5 &\geqslant 0 \\ x_1 \geqslant 0, \quad x_2 &\geqslant 0. \end{aligned} \tag{27.13}$$

Avem

$$\frac{\partial f_1}{\partial x_1} = 2x_1, \quad \frac{\partial f_1}{\partial x_2} = 8x_2$$

$$\frac{\partial f_2}{\partial x_1} = 32x_1 - 16, \quad \frac{\partial f_2}{\partial x_2} = 8x_2 - 4.$$

Sistemul (27.11) devine

$$\frac{x_1}{16x_1 - 8} = \frac{2x_2}{2x_2 - 1} = -\lambda.$$

De aici se deduce

$$x_1 = \frac{8}{16 + \lambda_1} \tag{27.14}$$

$$x_2 = \frac{1}{2(1 + \lambda_1)}, \quad \text{unde } \lambda_1 = \frac{1}{\lambda}.$$

Scanned with OKEN Scanner

Eliminînd parametrul λ_1 din (27.14), obţinem

$$30x_1x_2 + x_1 - 16x_2 = 0.$$

Din (27.14) se vede că pentru $\lambda_1 \to 0$ şi $\lambda_1 \to \infty$ se obţin respectiv punctele $C_2\left(\frac{1}{2}, \frac{1}{2}\right)$ şi $C_1(0, 0)$ care sînt tocmai centrele elipselor $f_2(\mathbf{x}) = 0$ şi $f_1(\mathbf{x}) = 0$.

Pentru a găsi curba eficientă, se introduc expresiile lui x_1 şi x_2 din (27.14) în sistemul (27.13) şi se deduce că $\lambda_1 \in [0, 4]$. Prin urmare ecuaţia curbei eficiente este

$$x_1 = \frac{8}{16 + \lambda_1}$$

$$x_2 = \frac{1}{2(1 + \lambda_1)}, \quad 0 \leqslant \lambda_1 \leqslant 4.$$

Graficul acestui arc este ilustrat în Figura 27.1.

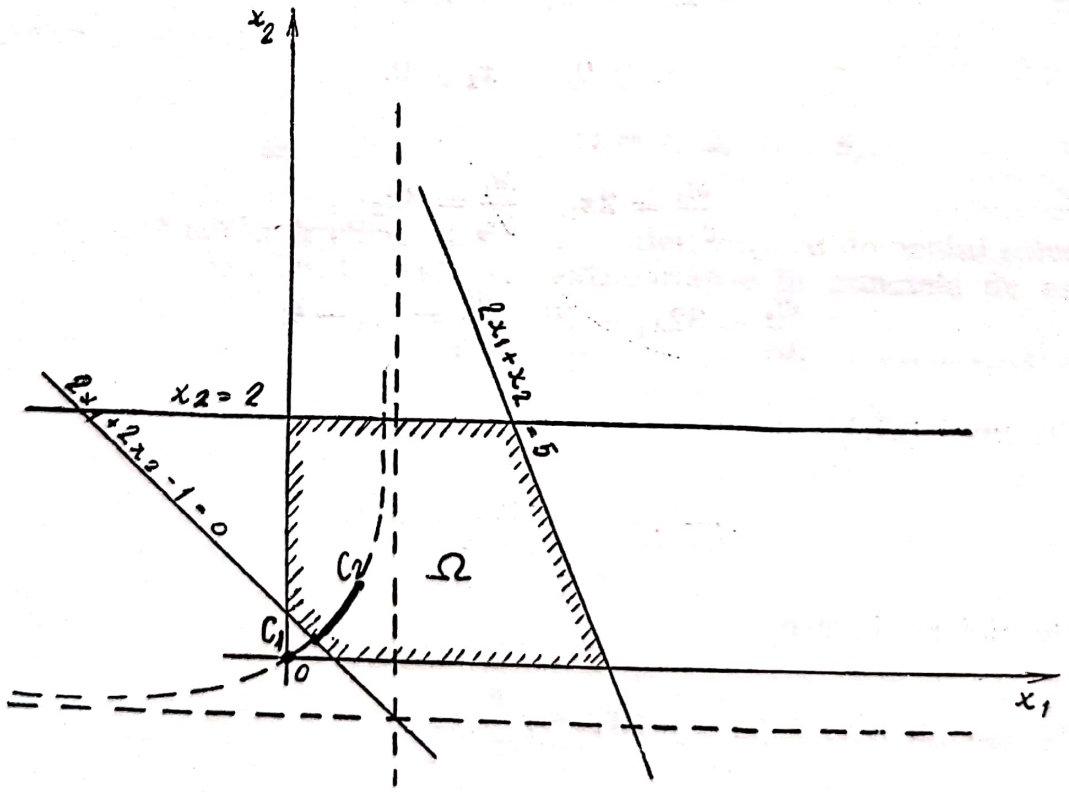

Figura 27.1.

Scanned with OKEN Scanner

Am văzut că în general există curbe de eficiență. Vom vedea că în cazul a cel puțin trei funcții de scop există și suprafețe de eficiență de diverse dimensiuni. Existența și determinarea acestor mulțimi de eficiență se bazează pe următorul rezultat [56]:

TEOREMA 27.3. *Condiția necesară și suficientă pentru ca* $\mathbf{x} \in \Gamma_q$ *să fie un punct eficient în raport cu* Y *este ca vectorii* ∇f_j $(j = 1, 2, \ldots, p)$ *și* $\mathbf{a}^k$ $(k = 1, 2, \ldots, q)$ *să fie pozitiv-liniar dependenți, adică*

$$\sum_{j=1}^{p} \lambda_j \nabla f_j(\mathbf{x}) + \sum_{k=1}^{q} \mu_k \mathbf{a}^k = 0, \tag{27.15}$$

unde

$$\lambda_j \geqslant 0 \quad \forall j \in \{1, 2, \ldots, p\}, \ \mu_k \geqslant 0, \quad \forall k \in \{1, 2, \ldots, q\}$$

și

$$\exists j_0 \in \{1, 2, \ldots, p\}, \quad \lambda_{j_0} > 0 \ .$$

Se vede că în cazul $q = 0$ (adică $\Gamma_0 = \operatorname{int} \Omega$), Teorema 27.3 conține ca un caz particular Teorema 27.2 (cînd există două constante $\lambda_j > 0$ în (27.15)).

În cazul cînd în (27.15) există trei constante pozitive și $q = 0$, adică

$$\lambda_{j_1} \nabla f_{j_1}(\mathbf{x}) + \lambda_{j_2} \nabla f_{j_2}(\mathbf{x}) + \lambda_{j_3} \nabla f_{j_3}(\mathbf{x}) = 0 \tag{27.16}$$

și $\lambda_j = 0$, $j \neq j_1, j_2, j_3$, atunci (27.16) definește o suprafață bidimensională $S_{j_1 j_2 j_3}$ din $\mathbf{R}^n$. Intersecția $S_{j_1 j_2 j_3} \cap \Omega$ este formată din puncte eficiente în raport cu Y, numită *suprafață de eficiență pentru* f_{j_1}, f_{j_2} și f_{j_3} *în raport cu* Y.

În același mod se pot defini suprafețe de eficiență de dimensiune trei ș.a.m.d.

Dacă în (27.15) există două constante λ_j pozitive și o constantă $\mu_k > 0$, de exemplu $\lambda_{j_1} > 0$, $\lambda_{j_2} > 0$, $\mu_{k_1} > 0$ și $\lambda_j = 0$, $j \neq j_1, j_2$, $\mu_k = 0$, $k \neq k_1$, atunci (27.15) se scrie:

$$\lambda_{j_1} \nabla f_{j_1}(\mathbf{x}) + \lambda_{j_2} \nabla f_{j_2}(\mathbf{x}) + \mu_{k_1} \mathbf{a}^{k_1} = 0. \tag{27.17}$$

Dacă la (27.17) se adaugă condiția

$$\mathbf{a}^{k_1} \mathbf{x} = b_{k_1}$$

se obține un sistem care definește o curbă $\Gamma_{j_1 j_2}^{k_1}$ situată pe hiperplanul $\mathbf{a}^{k_1} \mathbf{x} = b_{k_1}$. Intersecția $\Gamma_{j_1 j_2}^{k_1} \cap \Omega$ reprezintă de asemenea o curbă eficientă situată pe frontiera domeniului Ω.

Scanned with OKEN Scanner

În mod analog se pot obține curbele eficiente situate pe o muchie oarecare a domeniului Ω.

3. Un algoritm pentru puncte eficiente

Folosind criteriul dat în Teorema 27.1 pentru caracterizarea punctelor eficiente în raport cu Y se poate da un algoritm care permite găsirea unui punct eficient oricare ar fi Y.

Algoritmul constă din următorii pași:

1°. Se pornește cu un punct arbitrar $\mathbf{x}^0 \in \Omega$.

Să presupunem că s-au construit deja aproximațiile succesive $\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^k$. Presupunem că $\mathbf{x}^k \in \Gamma_q$, unde $q \in \{1, 2, \ldots, m\}$. Atunci se consideră sistemul (27.7) corespunzător punctului $\mathbf{x}^k$. Dacă acest sistem este incompatibil oricare ar fi vectorul nenul $\varepsilon \geqslant \mathbf{0}$, atunci punctul $\mathbf{x}^k$ este un punct eficient în raport cu Y. În caz contrar se procedează ca la punctul 2°.

2°. Se rezolvă problema de programare liniară

$$\max \mathbf{z} = \max \varepsilon \tag{27.18}$$

în condițiile

$$\begin{aligned} &\mathbf{g}^j(\mathbf{x}^k)\mathbf{u} - \varepsilon_j \geqslant 0, && (j = 1, 2, \ldots, p) \\ &\mathbf{a}^i\mathbf{u} \leqslant \mathbf{0}, && (i = 1, 2, \ldots, q) \\ &|u_k| \leqslant M, && (k = 1, 2, \ldots, n) \\ &\varepsilon_j \geqslant \varepsilon, && (j = 1, 2, \ldots, p) \end{aligned} \tag{27.19}$$

unde M este o constantă pozitivă suficient de mare.

Fie $\mathbf{u}^k$ soluția acestei probleme. Atunci se construiește următoarea aproximație succesivă $\mathbf{x}^{k+1}$ după cum urmează:

Fie

$$\tau_0^k = \min_{(j)} \{t_j \in \mathbf{R}^+ \mid \mathbf{a}^j(\mathbf{x}^k + t_j\mathbf{u}^k) = \mathbf{b}_j,\ j > q\}$$

$$\tau_i^k = \min \{t \in \mathbf{R}^+ \mid \min_{(i)} \mathbf{f}_i(\mathbf{x}^k + t\mathbf{u}^k)\}$$

și

$$\tau^k = \min (\tau_0^k, \tau_1^k, \ldots, \tau_p^k).$$

Scanned with OKEN Scanner

Atunci

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \tau^k \mathbf{u}^k.$$

Din modul de construcție a direcției $\mathbf{u}^k$ rezultă că

$$\forall k \in \mathbf{N} \Rightarrow f_i(\mathbf{x}^{k+1}) \leqslant f_i(\mathbf{x}^k), \qquad (i = 1, 2, \ldots, p)$$

și $\exists j_k \in \{1, 2, \ldots, p\}$ astfel încît

$$f_{j_k}(\mathbf{x}^{k+1}) < f_{j_k}(\mathbf{x}^k),$$

ceea ce arată că

$$\forall k \in \mathbf{N} \Rightarrow \mathbf{f}(\mathbf{x}^{k+1}) < \mathbf{f}(\mathbf{x}^k)$$

și prin urmare algoritmul este monoton.

Se demonstrează [56] că acest algoritm este convergent și că

$$\tilde{\mathbf{x}} = \lim_{k \to \infty} \mathbf{x}^k$$

este un punct eficient în raport cu Y.

Exemplul 27.2. Să se găsească un punct eficient în domeniul $\Omega \in \mathbf{R}^3$, definit de inegalitățile

$$\begin{aligned} x_1 + \quad x_2 + \quad x_3 - 1 &\geqslant 0 \\ -2x_1 - \quad 3x_2 - \quad x_3 + 8 &\geqslant 0, \\ x_1 \geqslant 0, \quad x_2 &\geqslant 0, \quad x_3 \geqslant 0 \end{aligned}$$

în raport cu

$$Y = \{\mathbf{y} \in \mathbf{R}^2 \mid \mathbf{y} = (y_1, y_2), \ \mathbf{x} \in \Omega\},$$

unde

$$\begin{aligned} y_1 = f_1(\mathbf{x}) &= 3x_1^2 + \quad x_2^2 + 4x_3^2 \\ y_2 = f_2(\mathbf{x}) &= 3x_1^2 + 4x_2^2 + \quad x_3^2. \end{aligned}$$

Scanned with OKEN Scanner

Se consideră ca punct de plecare $\mathbf{x}^0 = (1, 1, 1) \in \Omega$.

Avem

$$\mathbf{g}_0^1 = -\nabla f_1(\mathbf{x}^0) = -(6, 2, 8)$$
$$\mathbf{g}_0^2 = -\nabla f_2(\mathbf{x}^0) = -(6, 8, 2)$$

Prin urmare se rezolvă problema de programare liniară

$$\max \varepsilon$$

în condiţiile

$$\mathbf{g}_0^1\mathbf{u} - \varepsilon \geqslant 0$$
$$\mathbf{g}_0^2\mathbf{u} - \varepsilon \geqslant 0$$
$$|u_k| \leqslant 100, \qquad (k = 1, 2, 3).$$

Soluţia acestei probleme este $\mathbf{u}^0 = (-100, -100, -100)$. Evident că se poate lua $\mathbf{u}^0 = (-1, -1, -1)$. Atunci se găseşte că $\tau^0 = \frac{2}{3}$ iar noua aproximaţie

$$\mathbf{x}^1 = (1/3, 1/3, 1/3).$$

În acest punct

$$\mathbf{g}_1^1 = -(2, 2/3, 8/3), \; g_1^2 = -(2, 8/3, 2/3).$$

Noua problemă de programare este:

$$\max \varepsilon$$

în condiţiile

$$\mathbf{g}_1^1\mathbf{u} - \varepsilon \geqslant 0$$
$$\mathbf{g}_1^2\mathbf{u} - \varepsilon \geqslant 0$$
$$|u_k| \leqslant 100, \qquad (k = 1, 2, 3)$$

Soluţia acestei probleme (după simplificare) este $\mathbf{u}^1 = (2, -3, 1)$, iar $\tau^1 = 10/294$. Astfel că noua aproximaţie este

$$\mathbf{x}^3 = \mathbf{x}^1 + \tau^1\mathbf{u}^1 = (59/147, 34/147, 54/147).$$

Scanned with OKEN Scanner

Se constată că arcul de curbă

$$3x_1x_2 + 3x_1x_3 - 5x_2x_3 = 0$$

cuprins între punctele (4/19, 12/19, 3/19) și (4/19, 3/19, 12/19) este o curbă eficientă de pe planul $x_1 + x_2 + x_3 = 1$ și deoarece, punînd $F(\mathbf{x}) = 3x_1x_2 + 3x_1x_3 - 5x_2x_3$, avem

$$F(59/147,\ 34/147,\ 54/147) = \frac{602}{7203} \cong 0{,}08$$

se vede că șirul aproximațiilor succesive $\mathbf{x}^k$ converge către un punct eficient de pe această curbă eficientă pentru programul dat.

Scanned with OKEN Scanner

DES MÉTHODES POUR LA SOLUTION DU PROBLÈME DE LA PROGRAMMATION NON-LINÉAIRE

(RESUMÉ)

Dans ce travail on fait une synthèse des principaux résultats obtenus dans la solution effective des problèmes de la programmation non-linéaire. La plupart des résultats sont donnés sans démonstrations, cependant quelques-uns des plus importants sont présentés en détail.

Ainsi, après avoir fait un court historique du problème de la programmation linéaire et non-linéaire, on énonce le problème de la programmation non-linéaire et on donne les théorèmes de Kuhn-Tucker. Puis on présente quelques-unes des méthodes les plus connues pour la solution du problème de la programmation non linéaire (la méthode des sections planes ; la méthode des petits pas ; la méthode des grands pas ; la méthode des centres ; etc.) et aussi quelques méthods élaborées par les mathématiciens roumains. Suivent quelques méthodes spéciales pour la solution du problème de la programmation quadratique et pour le problème de la programmation segmentaire-linéaire.

A la fin du travail on énonce le problème de la programmation linéaire et non-linéaire pour plusieurs fonctions économiques et on donne quelques algorithmes pour ce problème.

Dans ce qui suit nous reproduisons quelques unes des méthodes ci-dessus rappelées.

La méthode des sections planes („Cutting plane-method")

Cette méthode a été présentée en trois variantes. La première a été donnée par CHENEY, E. W. et GOLSTEIN, A. A. [13] en 1959. La deuxième a été donnée en 1960 par KELLEY J. E. [35] et la troisième en 1963 par HARTLEY, H. O. and HOCKING, R. R. [26]. La variante qui suit est la deuxième. On considère le problème :

$$\min f(\mathbf{x}) = \min \mathbf{cx},$$
$$g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots, m)$$

Scanned with OKEN Scanner

où $\mathbf{c} = (c_1, c_2, \ldots, c_n)$, $\mathbf{x}' = (x_1, x_2, \ldots, x_n)$. On suppose que le domaine défini par les inégalités $g_j(\mathbf{x}) \leqslant 0$ est borné, c'est-à-dire il existe un polyèdre

$$P_0 = \{\mathbf{x} | \mathbf{A}\mathbf{x} \leqslant \mathbf{b}\},$$

où $\mathbf{A}$ est une matrice du type (m, n), qui contient Ω. De même on suppose que les fonctions g_j sont convexes et différentiables sur P_0. L'algorithme consiste dans les opérations suivantes: d'abord on résoud le problème de la programmation linéaire:

$$\min f(\mathbf{x}) = \min \mathbf{c}\mathbf{x}, \quad \mathbf{x} \in P_0.$$

Soit $\mathbf{x}^0$ la solution de ce problème et soit

$$g_r(\mathbf{x}^0) = \max_{(j)} g_j(\mathbf{x}^0).$$

Alors on définit le polyèdre $P_1 \subset \mathbf{R}^n$ comme l'intersection de P_0 et du domaine défini par l'inégalité

$$g_r(\mathbf{x}^0) + \nabla g_r(\mathbf{x}^0)](\mathbf{x} - \mathbf{x}^0) \leqslant 0.$$

Avec le polyédre P_1 on procède de la même manière. Ainsi on peut construire une suite $\mathbf{x}^k$ d'approximations successives et on montre que

$$\mathbf{x}^* = \lim_{k \to \infty} \mathbf{x}^k$$

est la solution du problème initial.

La méthode des grands pas

Cette méthode est une des méthodes les plus connues et elle a été donnée aussi dans plusieurs variantes. Les premières ont été données par FRISCH, R. A. K. [20], ROSEN, J. B. [67] et ZOUTENDIJK, G. [82]. Avec cette méthode on peut trouver la solution du problème suivant:

$$\max f(\mathbf{x}),$$

$$\mathbf{A}\mathbf{x} \leqslant \mathbf{b},$$

Scanned with OKEN Scanner

où f est une fonction concave qui a des dérivées partielles du premier ordre continues dans le domaine des solutions Ω. L'algorithme part d'un point arbitraire $\mathbf{x}^0 \in \Omega$ et puis on passe de l'itération $\mathbf{x}^k$ à l'itération

$$\mathbf{x}^{k+1} = \mathbf{x}^k + t_k \mathbf{s}^k,$$

où $\mathbf{s}^k$ est une direction admissible et t_k le pas de l'itération.

Diverses variantes de cette méthode diffèrent entre elles seulement par les procédés de détermination de la direction de „plongeon". Un procédé assez simple de détermination des ces directions a été donné aussi par MARUȘCIAC, I. et RĂDULESCU, M. dans [49–52].

La méthode des centres

Cette méthode a été élaborée en 1964 par HUARD, P. [29] et elle est fondée sur la détermination du centre d'un domaine convexe $\Omega \subset \mathbf{R}^n$ à l'aide d'une fonction-distance $\theta(\mathbf{x}, \Omega)$ convenablement définie. Si une telle fonction est définie alors on peut résoudre le problème

$$\max f(\mathbf{x})$$

$$g_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots, m),$$

où f est une fonction concave et g_j sont des fonctions convexes, à la manière suivante. On part d'un point arbitraire $\mathbf{x}^0 \in \Omega$ et puis on considère un domaine convexe défini comme suit:

$$\Omega_1 = \Omega \cap \{\mathbf{x} \in \mathbf{R}^n | f(\mathbf{x}) \geqslant f(\mathbf{x}^0)\}.$$

Soit $\mathbf{x}^1$ le centre de Ω_1, c'est-à-dire, pour lequel on a

$$\theta(\mathbf{x}^1, \Omega_1) = \max \{\theta(\mathbf{x}; \Omega_1) | \mathbf{x} \in \Omega_1\}.$$

Avec point $\mathbf{x}^1$ on procède de la même manière et ainsi on trouve le nouveau centre $\mathbf{x}^2$ et ainsi de suite. On montre que

$$\mathbf{x}^* = \lim_{k \to \infty} \mathbf{x}^k$$

est la solution du problème.

Scanned with OKEN Scanner

Programmation segmentaire-linéaire

La programmation segmentaire-linéaire est un problème de la programmation convexe dans lequel les fonctions f et g_j sont convexes et linéaires par morceaux. Plus précisément, si

$$L_j(\mathbf{x}) = \sum_{k=1}^{n} a_{jk}\, x_k - b_j, \qquad (j = 1, 2, \ldots, m)$$

sont des fonctions linéaires données, alors on appelle la fonction

$$F(\mathbf{x}) = \max_{(j)} L_j(\mathbf{x})$$

segmentaire-linéaire.

Par problème de la programmation segmentaire-linéaire on entend le problème :

$$\min F(\mathbf{x}),$$

$$G_j(\mathbf{x}) \leqslant 0, \qquad (j = 1, 2, \ldots, m),$$

où les fonctions F, G_j sont des fonctions segmentaires-linéaires.

Pour un tel problème on peut donner (voir [22]) un algorithme fini plus simple que les algorithmes connus pour le problème général de la programmation convexe.

Programmation monotone

Une fonction f définie sur un ensemble convexe $\Omega \subset \mathbf{R}^n$ s'appelle monotone sur Ω si

$$\forall \mathbf{x}', \mathbf{x}'' \in \Omega \text{ et } \forall t \in [0,1] \Rightarrow \min [f(\mathbf{x}'), f(\mathbf{x}'')] \leqslant f(\mathbf{x}) \leqslant \max [f(\mathbf{x}'), f(\mathbf{x}'')]$$

où

$$\mathbf{x} = t\mathbf{x}' + (1 - t)\mathbf{x}''.$$

Un problème de programmation mathématique s'appelle monotone si la fonction économique f est monotone sur l'ensemble convexe Ω des solutions admissibles.

Scanned with OKEN Scanner

Un cas important de la programmation monotone présenté est celui où la fonction économique f est une fonction rationnelle de la forme

$$f(\mathbf{x}) = \frac{c_1x_1 + c_2x_2 + \ldots + c_nx_n + \alpha}{d_1x_1 + d_2x_2 + \ldots + d_nx_n + \beta} = \frac{\mathbf{cx} + \alpha}{\mathbf{dx} + \beta},$$

où $\mathbf{dx} + \beta > \mathbf{0}$, $\mathbf{x} \in \Omega$.

Deux algorithmes sont donnés pour ce problème. L'un appartient à KASANYUK, S. A. et KUCHER, B, N., [34] et l'autre appartient à KUCHER, B. M. [39].

La problème de la programmation mathématique à plusieurs fonctions économiques.

Il y a beaucoup de problemès économiques qui conduisent au probleme de programmation mathématique dans lequel il faut minimiser ou maximiser simultanément plusieurs fonctions économiques. Dans le cas où tant les fonctions économiques que les restrictions sont linéaires, le problème a été étudié par ZUHOVICZKII, S. I. [83] et REMES, E. I., STEIN — BERG, A. S. [66]. Un algorithme a été construit aussi par MARUŞCIAC, I., RĂDULESCU, M. [54] pour le problème analogue, où les fonctions économiques sont quadratiques et les restrictions sont linéaires.

Une autre manière de poser le problème de la programmation à plusieurs fonctions économiques a été formulé pour la première fois par CHARNES, A. et COOPER, W. [10]. Le problème consiste dans la determination des points, appelés „points efficients" du domaine des solutions admissibles, qui optimisent simultanément plusieurs fonctions économiques.

On considère le vecteur-fonction

$$\mathbf{y} = \mathbf{f}(\mathbf{x}),$$

où

$$\mathbf{y}' = (y_1, y_2, \ldots, y_p),\ \mathbf{x}' = (x_1, x_2, \ldots, x_n) \text{ et}$$

$$y_j = f_j(\mathbf{x}) = f_j(x_1, x_2, \ldots, x_n), \qquad (j = 1, 2, \ldots, p)$$

sont des fonctions continues et convexes dans

$$\Omega = \{\mathbf{x} \in \mathbf{R}^n \mid \mathbf{Ax} \leqslant \mathbf{b}\}.$$

Soit

$$Y = \{\mathbf{y} \in \mathbf{R}^p \mid \mathbf{y} = \mathbf{f}(\mathbf{x}),\ \mathbf{x} \in \Omega\}$$

Scanned with OKEN Scanner

En introduisant dans l'ensemble Y la relation d'ordre naturel on dit que $\mathbf{x}^0 \in \Omega$ est un point efficient par rapport à Y s'il n'existe aucun point $\mathbf{x} \in \Omega$ comparable à $\mathbf{x}^0$ par $\mathbf{f}$ tel que

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) < \mathbf{y}^0 = \mathbf{f}(\mathbf{x}^0).$$

On donne [56] des caractérisations des points efficients qui permetent ensuite d'élaborer des méthodes pour déterminer des points ou des sous-ensembles de points efficients.

Scanned with OKEN Scanner

BIBLIOGRAFIE

1. Abadie, I., *Nonlinear Programming*, Interscience publishers, North-Holland Publishing Comp., New York, 1967.
2. Anand Promila, Swarup Kanti., *Minimization of an indefinite quadratic function with linear constraints*, Ekonomicky Matematicky obzor **6**, nr. 4, 1970, pp. 381—387.
3. Arrow, K. J., Hurwicz L., *Reduction of constrained maximo to saddle-point problems*, Proceedings of the third Berkeley Symposium on Math. Stat. and Prob. Berkeley and Los Angeles: University of California Press, 1955.
4. Arrow, K. J., Hurwicz L., and Uzawa H., *Studies in linear and non-linear programming*, Stanford University Press, 1958.
5. Barankin, E., Dorfarnan R., *On Quadratic Programming*, University of California Press, Berkeley, 1958.
6. Beale, E. M. L., *On Optimizing a Convex Function Subject to Linear Inequalities*, Journal of Royal Statistical Society, nr. 17, 1955.
7. Bód, P., *Lineáris programozás több egyidejuleg adott célfügvény szerint*, Magy. Tud. Akad. intézet Közlem., VIII, b. 4. pp. 541—556, 1963.
8. Boot, G. G., *Quadratic Programming*, North Holland co., Amsterdam, 1964.
9. Candler, W., Townsley, R. J., *The maximization of quadratic function of variables subject to linear inequalities*, Management Science, **10**, pp. 515—523, 1964.
10. Charnes, A., and Cooper, W. W, *Management models and industrial applications of Linear Programming*, Management Science **4**, pp. 38—92, 1957.
11. Charnes A., Cooper W. W., *Programming with Linear Fractional Functionals*, Naval Research, Logistics Quarterly **9**, nr. 3—4, pp, 181—187, 1962.
12. Charnes A, Lemke, C., *Minimization of Nonlinear Separable Convex Functionals*, Naval Research Logistics Quarterly nr. 1, 1954.
13. Cheney, E. W., and Goldstein, A. A., *Newton's method for convex programming and Tchebycheff approximation*, Numer. Math. Bd. 1., 1959.
14. Collatz, L. and Wetterling, W., *Optimierungsaufgaben*, Springer-Verlag, 1966.
15. Dantzig, G. B. and Wolfe, Ph., *The decomposition algorithm for linear programming*, Econometrica, Vol. **29**, 1961.
16. Dorn, W. S., *Non-linear programming* — A survey., Management Science, vol. **9**, 1963.
17. D'Esopo, D. A., *A Convex Programming Procedure*, Naval Research Logistics Quarterly, nr. 6, 1959.
18. Farkas, J., *Über die Theorie der einfachen Ungleichungen*, Journal für die Reine und Angewandte Mathematik, Bd. 124, 1902.
19. Fiacco, A. V., and McCormick, G. P., *Programming under monlinear constraints by unconstrained minimization : A primal-dual method*, RAC-TP-96, Reasearch Analysis Corporation, 1963.
20. Frisch, R. A. K., *The multiplex method for linear and quadratic programming*, Mem. Univ. Social. Institute of Oslo, 1957.
21. Goldman, A. G., *Sb. Lineinye neravenstva i smežnye vaprosy* J. L., 1959.

Scanned with OKEN Scanner

22. Golstein, E. G., Iudin, D. B., *Novie napravlenia v lineinom programmirovanii*, Izd. Sovet. Rodina, Moskva, 1966.
23. Graves, R. L., Wolfe, Ph., *Recent advances in Mathematical Programming*, Mec. Graw-Hill Book Co., London, 1963.
24. Griffith, R. E., and Stewart, R. A., *A nonlinear programming technique for the optimization of continuous prossessing systems*, Management Science, vol. **7**, 1961.
25. Hadley, G., *Nonlinear and Dynamic Programming*, *Addison-Wesley*. Publishing Co., London, 1959.
26. Hartley, H. O. and Hocking, R. R., *Convex programming by tangential approximation*, Management Science., vol. **9**, 1963.
27. Hestenes, M. R. and Stiefel, E., *Method of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards vol. **49**, 1952.
28. Houthakker, H., *The Capacity Method of Quadratic Programming*, Econometrica nr. 28, 1960.
29. Huard, P., *Résolution de programmes mathématiques à contraintes non linéaires par la méthode des centres*, Note Electricité de France, 1964.
30. Joksh, M. C., *Programming with Fractional Linear Objective Functions*, Naval Research Logistics Quarterly, **11**, nr. 3—4, pp. 197—204, 1964.
31. Kantorovici, L. V., *Matematiceskie metodî v organizații i planirovanii proizvodstva*, Izd. L. G. U., 1939.
32. — , *Ob odnon effectivnon metode reșenia nekotorih classov ekstremalnîh problem*, D.A.N., SSSR, **28**, nr. 3, pp. 212—215, 1940.
33. Kantorovici, L. V. și Gavurin, M. K., *Primenenie matematiceskih metodov v voprosah analiza gruzopotokiv*. Sb. „Problemî povișenia effectivnosti rabotî transporta", Izd. Akad. N. SSSR, pp. 110—138, 1940.
34. Kasanyuk, S. A., Kucher, B. M., *Pro monotonne programirovania*, Dopovidi Akad. Nauk, URSR **1**, pp. 18—21, 1967.
35. Kelley, J. E., *The cutting plane method for solving convex programs*, J. Soc. Indus. Appl. Math., vol. **8**, 1960.
36. Kleibohm, K., *Ein Verfahren zur approximativen Lösung von konvexen Programmen*, Diss. Zürich, 1966.
37. Krelle, W., *Preistheorie*, J. C., Mohr Tübingen und Polygraphischer Verlag, Zürich, 1961.
38. Kolumbán, I., *Über die Characterisierung der Lösungen eines optimalen Approximations problem*, Mathematica (Cluj), **10** (33), **2**, pp. 285—291, 1968.
39. Kucher, B. M., *Pro oden algoritm rișennia zadaci monotonogo programuvannia*, Dopovidi Akad. Nauk, URSR., **10**, pp. 869—872, 1967.
40. Kuhn, H. W. and Tucker, A. W., *Non-linear programming*. In: Proc. of the Second Berkeley Symposium on Math. Stat. and Probab., University of California Press, 1950.
41. Künzi, H. P., *The duoplex method in nonlinear programming*, SIAM Journ. of Control, vol. **4**, 1966.
42. Künzi, H. P. und Krelle, W., *Nichtlineare Programmierung*, Springer-Verlag, 1962.
43. Küntzi, H. P., Kall, P., Oettli, W., *A contribution to indefinite quadratic programming*, Technical discussion paper, I.B.M. Zürich Research Laboratory Ruschlikon, Switzerland, 1964.
44. Küntzi, H. P., *Zum Heutigen der nichlinearen Optimierungstheorie*, Unternehmensforschung, Bd. **12**, Heft, 1, pp. 1—22, 1968.
45. Maliță, M., Zidăroiu, C., *Matematica organizării*, Ed. Tehnică, 1971.
46. Martos B., *The direct power of Simplicial method in continuous programming*, Preliminary paper submitted to the international Simposium on Mathematical Programming, London, 1964,
47. Martos, B., *Hyperbolyc Programming*, Naval Research Logistics Quarterley, **11**, pp. 135—156, 1964.
48. Maruşciac, I., Munteanu, E. și Rus, T., *Rezolvarea problemei deschiderii optime a unui sistem de puncte materiale cu ajutorul unei mașini electronice de calcul*, Studii și Cercet. Mat. (Cluj) t. **13**, pp. 105—113, 1962.
49. Maruşciac, I. și Rădulescu, M., *Metod nailucișevo spuska vo vîpuklom programmirovanii*, Mathematica (Cluj), t. **6**, (29), nr. 2, pp. 264—271, 1964.

Scanned with OKEN Scanner

50. Maruşciac, I., *Un algoritm pentru rezolvarea unui sistem de inegalităţi liniare*, Studia Univ. „Babeş-Bolyai", Ser. I, Fasc. 1, pp. 51—57, 1965.
51. — , *Une algorithme pour résoudre le problème de la programmation quadratique*, Mathematica (Cluj), t. **7**, (30), nr. 2, pp. 287—296, 1965.
52. — , *O metodă de rezolvare a problemei de programare convexă*, Studia Univ. „Babeş-Bolyai", Ser. I., fasc. 1., pp. 41—52, 1966.
53. — , *O formă explicită a soluţiei problemei programării pătratice*. Studia Univ. „Babeş-Bolyai", Ser. Math. Phis., Fasc. 2, pp. 59—66, 1968.
54. Maruşciac, I. şi Rădulescu, M., *Un problème de la programmation quadratique à plusieurs fonctions économiques*, Studia Univ. „Babeş-Bolyai", Ser, I, fasc. 1,pp. 103—111, 1970.
55. — , *Un problème général de la programmation Linéaire aux plusieurs fonctions économiques*, Studia Univ. „Babeş-Bolyai", Ser. I, Fasc. 2, pp. 55—65, 1970.
56. — , *Sur l'ensemble des points efficients d'un problème de la programmation Mathematiques*, Anal. Univ. Al. I. Cuza Iaşi, t. **18**, fasc. 2, pp. 209—226, 1972.
57. Mihoc Gh., Ionescu, H., *Bazele matematice ale programării liniare*, Ed. Ştiinţifică, Bucureşti, 1965.
58. Mihoc, Gh., şi Nădejde Ileana, *Programarea matematică*, Programarea parametrică şi neliniară. Ed. Ştiinţiţcă, 1966.
59. Mocanu, C., *Observaţii asupra rezolvării problemelor de programare pătratică*, Stud şi Cercet. Mat. Cluj, t. XIV, nr. 2, 1963.
60. — , *Sur la résolution des problèmes de programmation quadratique*, Mathematica (Cluj), vol. **6**, (29), pp. 51—61, 1964.
61. Motanov, V. G., *O zadaciah optimizaţii s drobnolineinym funkţionalom*, Ekonom. i Matem. Metody, **7**, nr. 4, pp. 586—592, 1971.
62. Motzkin, T. S., *Beiträge zur Theorie der linearen Ungleichungen*, Diss. Basel, 1933.
63. Munteanu E., Radó F., *Calculul şarjelor celor mai economice la cuptoarele de topit fontă*, Studii şi Cercet. Matem. Cluj, Fasc. anexă, t. **11**, pp. 149—158, 1960.
64. Neumann, J. von, *Zur Theorie der Gesellschaftsspiele*, Math. Ann., Bd. 100, 1928.
65. Orden, A., *Stationary point of quadratic maximum problem*, Computer Journal, vol. **3**, nr. 7, 1964.
66. Remez, E. I. şi Steinberg, A. S., *Pro deiaki ekstremalni zadaci uzagalnenocebîsovscovo tipu ta pro metod zrivnialnîh spuschiv*, Dopovidi A.N.URSR. nr. 9, pp. 983—987, 1961.
67. Rosen, J. B., *The gradient projection method for nonlinear programming*, SIAM, J., vol. **8**, 1960.
68. Simonard, M., *Programmation linéaire*, Dunod, Paris, 1962.
69. Stiefel, E., *Note on Jordan elimination, linear programming and Tchebycheff approximation*, Numerische Mathematik, vol. **2**, pp. 1—17, 1960.
70. Stiemke, E., *Über positive Lösungen homogener linearer Gleichungen*, Math. Ann. Bd. **76**, 1915.
71. Swarup Kanti., *Programming with indefinite quadratic function with linear constraints*, Cahiers de Centre d'Etudes de Rech. Op. **8**, nr. 2, pp. 132—136, 1966.
72. — , *Quadratic programming*, Cahiers de Centre d'Etudes de Rech. Op. **8**, nr. 4, pp. 223—234, 1966.
73. — , *Fractional Programming with nonlinear constraints*, ZAMM 46, 1966.
74. — , *Duality for Transportation Problem in fractional Programming*, Cahiers d'Etudes de Rech. Op. 10, nr. 1, 1968.
75. Wagner, H. M., Yuan, J. S. C., *Algorithmic equivalence in linear fractional programming*, Manag. Sc. 14, pp. 301—306, 1968.
76. Wolfe, Ph., *The simplex Method for Quadratic Programming*, Econometrica nr. 27, 1959.
77. — , *Accelerating the cutting plane method for nonlinear programming*, J. Soc. Indust. Appl. Math. vol. **9**, 1961.
78. — , *Methods of nonlinear programming*, In Recent advance in mathematical Programming, R. L. Graves and Ph. Wolfe (eds), McGraw-Hill, 1963.
79. — , *Method of nonlinear programming*. In: Nonlinear Programming, J. Abadie (ed), North-Holland Publ. Co., 1967.
80. Zmorovici, V. A., *O nekotoryh variaczionnyh zadaciah teorii odnolistnyh funczii*, Ucrain. Mat. Journal, **4**, pp. 276—296, 1952.

Scanned with OKEN Scanner

81. Zoutendijk, G., *Nonlinear Programming*, SIAM Journal on Control. vol., **4**, 1966.
82. —, *Method of feasible directions*, Elsevier, Amsterdam, 1960.
83. Zuhoviţki S. I., *Algoritm dlia reşenia odnoi obboşcionnoi zadacii lineinovo programmirovania*, DAN. SSSR, t. **133**, nr. 1, pp. 20—23, 1960.
84. Zuhoviţki S. I., şi Leifman L. I., *O vîcislitelnoi scheme odnovo algoritma dlia vîpuklovo kvadraticinovo programmirovania*, S.O. DAN. SSSR, Novosibirsk, 1962.
85. Zuhoviţki S. I., Poliak R. A. şi Primak M. I., *Algoritm dlia reşenia zadacii vîpuklovo cebîşevskovo priblijenia*, DAN, SSSR, t. **151**, nr. 1, pp. 991—994, 1963.
86. Zuhoviţki S. I. şi Avdeeva L. I., *Lineinoie i vîpukloe programmirovanie*, Izd. Nauka, Moskva, 1964.

Scanned with OKEN Scanner

CUPRINS

CAPITOLUL I: **Problema generală de programare matematică**

§ 1. Introducere 5
§ 2. Istoricul problemei 6
§ 3. Formularea problemei generale de programare neliniară și ideile lui Kuhn și Tucker 7
§ 4. Metoda eliminării a lui Jordan, cu aplicații 13
§ 5. Metoda simplex 19

CAPITOLUL II: **Metode generale de rezolvare a problemelor de programare neliniară**

§ 6. Metoda planelor secante (Cutting—plane Method) 29
§ 7. Metoda pașilor mici 37
§ 8. Metoda pașilor mari 42
§ 9. Metoda gradientului redus 64
§10. Metoda centrelor a lui Huard 67
§11. Metoda SUMT a lui Fiacco și McCormick 73
§12. Metoda lui Kleibohm 74

CAPITOLUL III: **Cazuri speciale de programare neliniară**

§13. Minimizarea sumei modulelor funcțiilor liniare 79
§14. Aproximarea uniformă în domeniul complex 80
§15. Punctul cel mai apropiat de un sistem de plane 81
§16. Programarea segmentar-liniară 83
§17. Programarea monotonă 90

CAPITOLUL IV: **Programarea pătratică**

§18. Algoritmul lui Beale-Zuhovițki 100
§19. Reprezentarea funcției de scop ca sumă de pătrate 108

Scanned with OKEN Scanner

§20. Metoda funcţiilor omogene în rezolvarea problemelor de programare pătratică 110
§21. O metodă a ortogonalizării 113
§22. O formă explicită a soluţiei problemei de programare pătratică 116
§23. Optimizarea funcţiilor pătratice nedefinite 122
§24. Optimizarea unui produs de funcţii liniare. 131

CAPITOLUL V: **Programarea simultană a mai multor funcţii de scop**

§25. Programarea pătratică a mai multor funcţii de scop 146
§26. O problemă de programare liniară generală cu mai multe funcţii de scop 148
§27. Structura mulţimii punctelor de eficienţă a mai multor funcţii de scop . . 151

Rezumat în limba franceză 164

Bibliografie 170

Scanned with OKEN Scanner

Redactor: M. CREMENE; Tehnoredactor: A. MOLNÁR; Apărut: 1973; Comanda nr.: 1036; Coli de tipar: 11; Tiraj: 3800+140 ex. broşate; Hîrtia: velină de 80 g/m²; Format: 16/70×100. C. Z. pentru biblioteci: 519.283; Tiparul executat sub comanda nr. 873/1973. la Întreprinderea Poligrafică Cluj, str. Brassai nr. 5–7; Republica Socialistă România

Scanned with OKEN Scanner